



PUBLIC  
2024-12-12

# SAP Business Technology Platform

# Content

<b>1</b>	<b>SAP Business Technology Platform</b>	<b>7</b>
<b>2</b>	<b>Basic Platform Concepts</b>	<b>10</b>
2.1	Solutions and Services.	16
2.2	Regions.	17
	Regions and API Endpoints Available for the Cloud Foundry Environment.	20
	Regions and API Endpoints for the ABAP Environment.	37
	Regions for the Kyma Environment.	41
	Resilience, High Availability, and Disaster Recovery.	45
2.3	Environments.	47
	Cloud Foundry Environment.	49
	ABAP Environment.	65
	Kyma Environment.	70
	Neo Environment.	81
2.4	Trial Accounts and Free Tier.	82
2.5	Enterprise Accounts.	84
	Commercial Models.	86
	Using Free Service Plans.	91
2.6	Account Model.	94
2.7	Entitlements and Quotas.	100
2.8	User and Member Management.	104
	Platform Users.	105
	Business Users.	119
2.9	Tools, Programming Models, Programming Languages, and APIs.	120
	Tools.	121
	Programming Languages.	123
	Programming Models.	124
	Continuous Integration and Delivery (CI/CD).	125
	APIs.	128
	Cloud Management Tools — Feature Set Overview.	129
	Prerequisites and Restrictions.	141
<b>3</b>	<b>Getting Started</b>	<b>143</b>
3.1	Getting a Global Account.	144
3.2	Getting Started in the Cloud Foundry Environment.	147
	Getting Started with a Trial Account in the Cloud Foundry Environment.	147
	Getting Started with an Enterprise Account in the Cloud Foundry Environment.	158

3.3	Getting Started in the ABAP Environment. . . . .	165
	Getting Started with a Customer Account in the ABAP Environment. . . . .	166
	Getting Started with Custom Code Analysis in the ABAP Environment. . . . .	222
3.4	Getting Started in the Kyma Environment. . . . .	223
	Getting Started with an Enterprise Account in the Kyma Environment. . . . .	225
	Getting Started with a Trial Account in the Kyma Environment. . . . .	226
<b>4</b>	<b>Development. . . . .</b>	<b>231</b>
4.1	Development in the Cloud Foundry Environment. . . . .	233
	Best Practices. . . . .	236
	Developing Your First Application. . . . .	238
	Developing Applications and Services. . . . .	238
	Developing User Interface. . . . .	387
	Consuming APIs. . . . .	498
	Adding Authentication and Authorization. . . . .	499
	Setting Up Database Artifacts. . . . .	567
	Deploying to the Cloud Foundry Environment. . . . .	569
4.2	Development in the ABAP Environment. . . . .	705
	Getting Started as a Developer in the ABAP Environment. . . . .	706
	ABAP Development Tools for Eclipse: User Guides. . . . .	710
	ABAP Keyword Documentation. . . . .	711
	ABAP RESTful Application Programming Model. . . . .	711
	Assuring Quality of ABAP Code. . . . .	713
	ABAP Lifecycle Management. . . . .	715
	Identity and Access Management (IAM). . . . .	797
	Business Configuration. . . . .	863
	Integration and Connectivity. . . . .	881
	SaaS Applications and Multitenancy. . . . .	1297
	Released Components and Objects. . . . .	1426
	UI Development. . . . .	1830
	ABAP Service Instance URL. . . . .	1863
4.3	Development in the Kyma Environment. . . . .	1864
	Kyma Modules. . . . .	1866
	Access a Kyma Instance Using kubectl. . . . .	1973
	Using SAP BTP Services in the Kyma Environment. . . . .	1975
	Deploy Workloads in the Kyma Environment to Extend SAP Systems. . . . .	1977
	Expose and Secure a Workload with a JSON Web Token. . . . .	1979
	Develop Resilient Applications in the Kyma Runtime. . . . .	1983
	Choose a Backend for Kyma Eventing. . . . .	1984
	Available Technology. . . . .	1987
<b>5</b>	<b>Extensions. . . . .</b>	<b>1989</b>

5.1	Maintaining Unified Customer Landscape. . . . .	1990
5.2	Extensibility Concepts. . . . .	1992
5.3	Registering and Deregistering Systems. . . . .	1997
	Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP. . . . .	2002
	Register an SAP Marketing Cloud System in a Global Account in SAP BTP. . . . .	2006
	Register an SAP SuccessFactors System in a Global Account in SAP BTP. . . . .	2009
	Register an SAP Customer Experience System. . . . .	2014
	Registering a Third-Party System. . . . .	2016
	Deregistering or Removing a System. . . . .	2019
	Merging SAP Systems. . . . .	2021
5.4	Extending SAP Solutions . . . . .	2022
	Declaring System APIs and Events as Dependencies for Business Scenarios. . . . .	2024
	Including Systems in a Formation. . . . .	2027
	Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment. . . . .	2046
	Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment. . . . .	2108
	Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment. . . . .	2112
	Extending SAP Customer Experience Products in the Kyma Environment. . . . .	2136
<b>6</b>	<b>Administration and Operations. . . . .</b>	<b>2138</b>
6.1	Account Administration. . . . .	2139
	Account Administration in the Cockpit. . . . .	2140
	Account Administration Using Joule (Beta). . . . .	2318
	Account Administration Using the SAP BTP Command Line Interface (btp CLI) . . . . .	2321
	Account Administration Using APIs. . . . .	2378
	Account Administration Using Infrastructure as Code. . . . .	2420
6.2	Administration and Operations in the Cloud Foundry Environment. . . . .	2422
	Org Management Using the SAP BTP Command Line Interface (btp CLI). . . . .	2423
	About User Management in the Cloud Foundry Environment. . . . .	2430
	About Roles in the Cloud Foundry Environment. . . . .	2431
	Org Administration Using the Cockpit. . . . .	2433
	Org Administration Using the Cloud Foundry CLI. . . . .	2451
	Application Operations in the Cloud Foundry Environment. . . . .	2490
	Audit Logging in the Cloud Foundry Environment. . . . .	2501
6.3	Administration and Operations in the ABAP Environment. . . . .	2536
	SAP Fiori Apps in the ABAP Environment. . . . .	2538
	Communication Operations. . . . .	2843
	Technical Operations. . . . .	2926
	Troubleshoot Custom Apps. . . . .	2988
6.4	Administration and Operations in the Kyma Environment. . . . .	2991
	Create a Kyma Instance. . . . .	2992
	Managing Kyma Runtime Using the Provisioning Service API. . . . .	2994
	Available Plans in the Kyma Environment. . . . .	2998

Provisioning and Updating Parameters in the Kyma Environment . . . . .	3002
Add and Delete a Kyma Module. . . . .	3012
Access Kyma Application Logs. . . . .	3014
Kyma Environment Backup. . . . .	3015
Kyma Modules' Sizing. . . . .	3016
Change Storage Size in Kyma. . . . .	3019
<b>7 Security. . . . .</b>	<b>3021</b>
7.1 SAP Authorization and Trust Management Service. . . . .	3023
What Is the SAP Authorization and Trust Management Service? . . . . .	3030
Web Access Control. . . . .	3034
API Access Control. . . . .	3035
Authorization Entities. . . . .	3036
Monitoring and Troubleshooting. . . . .	3038
Security Considerations for the SAP Authorization and Trust Management Service. . . . .	3074
Configuration Options for the SAP Authorization and Trust Management Service. . . . .	3078
Third-Party Cookies and SAP Authorization and Trust Management Service. . . . .	3080
Security Recommendations for SAP Authorization and Trust Management Service. . . . .	3081
Rate Limiting. . . . .	3083
Limits for Technical Artifacts of the SAP Authorization and Trust Management Service. . . . .	3088
Configuring Backup. . . . .	3090
Accessibility Features in SAP Authorization and Trust Management Service. . . . .	3090
7.2 Default Role Collections of SAP BTP. . . . .	3090
7.3 Trusted Certificate Authentication. . . . .	3094
7.4 Principal Propagation. . . . .	3095
Principal Propagation from the Neo to the Cloud Foundry Environment. . . . .	3095
Principal Propagation from the Cloud Foundry to the Neo Environment. . . . .	3102
7.5 Data Protection and Privacy. . . . .	3109
Glossary for Data Protection and Privacy. . . . .	3111
Change Logging and Read-Access Logging. . . . .	3113
Personal Data Record. . . . .	3113
Deletion. . . . .	3114
Consent. . . . .	3120
7.6 Security in the Kyma Environment. . . . .	3120
Kyma Security Concepts. . . . .	3120
Secure Development in the Kyma Environment. . . . .	3126
Secure Administration and Operations in the Kyma Environment. . . . .	3132
<b>8 Getting Support. . . . .</b>	<b>3148</b>
Procedure. . . . .	3149
8.1 Providing Details for SAP HANA Service Database Problems. . . . .	3150
8.2 Gather Support Information. . . . .	3151

8.3	Platform Updates and Notifications. . . . .	3152
8.4	Operating Model. . . . .	3153
	Operating Model in the Cloud Foundry Environment. . . . .	3154
	Operating Model in the Kyma Environment. . . . .	3160
	Comparison between the Operating Models of Kyma and Cloud Foundry Runtimes. . . . .	3167
8.5	Support Components. . . . .	3180
<b>9</b>	<b>Glossary. . . . .</b>	<b>3265</b>

# 1 SAP Business Technology Platform

SAP Business Technology Platform (SAP BTP) is an integrated offering comprised of the following technology portfolios: application development; process automation; integration; data, analytics, and enterprise planning; artificial intelligence. The platform offers users the ability to turn data into business value, compose end-to-end business processes, connect entire IT landscapes, and personalize, build and extend SAP applications. This reduces the overall total cost of ownership maintaining SAP landscapes and third-party software across end-to-end business processes.

## SAP BTP in the Intelligent Enterprise

Nowadays, companies need access to live data to make informed decisions in real time and apply advanced technologies and best practices within agile, integrated business processes. That's why a key part of SAP's strategy is integrating end-to-end processes – whether the solutions are from SAP, from partners, or from third parties.

At SAP, we drive integration across our solution portfolio, including the following business scenarios: lead to cash, source to pay, design to operate, and hire to retire.

The SAP Business Technology Platform provides integration capabilities to ensure connected end-to-end business processes across SAP and third-party applications. The following key aspects, called Suite Qualities, facilitate a consistent experience across applications:

- Seamless User Experience: SAP Fiori can be used to gain a common look and feel of many SAP solutions, which improves user experience and reduces interruptions.  
See: [SAP Fiori](#).
- Consistent Security and Identity Management: SAP Cloud Identity services on SAP BTP enable you to manage identities and use the single sign-on capability across selected end-to-end processes.  
See: [SAP Cloud Identity Services](#) on SAP BTP.
- Aligned Domain Models, APIs and Events: Master Data Integration is a cloud service for master data integration. It provides a consistent view on master data across a hybrid landscape. You can use APIs for process integration among SAP solutions, SAP, and third parties, or by adopting an event-based integration strategy – all based on predefined integration content available on SAP API Business Hub.  
See: [SAP One Domain Model](#), [SAP Master Data Integration service](#), [SAP Integration Suite](#) on SAP BTP
- Embedded Analytics across Solutions: The analytical insights in various SAP solutions helps you make informed decisions. Embedded analytics from SAP Analytics Cloud are available e.g., in SAP S/4HANA and SAP SuccessFactors.  
See: [SAP Analytics Cloud](#) on SAP BTP.
- One Workflow Inbox: With the unified view of pending tasks across SAP solutions in both mobile and desktop environments, you can complete tasks faster.  
See: [SAP Task Center](#) on SAP BTP, [SAP Mobile Start](#). For full workflow modeling and visibility capabilities leverage, see: [SAP Process Automation](#) on SAP BTP
- Coordinated Lifecycle Management: Harmonized provisioning, setup and operations, and monitoring solutions allow you to optimize implementation projects and reduce manual integration configuration

effort. Automated provisioning and guided integration setup are available for selected scenarios in lead to cash, source to pay, recruit to retire, and design to operate.

See: [SAP for Me](#), [SAP CALM](#)

- End-to-End Process Blueprints: Process blueprints that follow the Industry Reference Architecture standard can help you reduce planning effort for implementation projects and also support architecture decisions.

See: [SAP API Business Hub](#)

Related information:

- [SAP's vision for the Intelligent Enterprise](#)
- [SAP's Integration Strategy whitepaper](#)
- [SAP Integration Strategy community](#)

## About This Guide

Get an overview of the basic platform concepts of SAP BTP and how they relate to each other. Learn how to manage your accounts and how to develop, extend, administer, and secure your cloud setup and applications on the platform.

### → Tip

The English version of this guide is open for contributions and feedback using GitHub. This allows you to get in contact with responsible authors of SAP Help Portal pages and the development team to discuss documentation-related issues. To contribute to this guide, or to provide feedback, choose the corresponding option on SAP Help Portal:

-  [Create issue](#): Provide feedback about a documentation page. This option opens an issue on GitHub.
-  [Edit page](#): Contribute to a documentation page. This option opens a pull request on GitHub.

You need a GitHub account to use these options.

More information:

- [Contribution Guidelines](#)
- [Introduction Video](#)
- [Introduction Blog Post](#)

More resources that you might find interesting:

## Best Practices

Get onboarded to SAP BTP. Learn about basic concepts and see our best practices for your development projects, from setting up the correct organizational structure to creating an account and security model, to developing and operating applications: [SAP BTP Administrator's Guide](#).

## **Services and Solutions**

Get an overview on the availability of SAP BTP solutions and services according to region, infrastructure provider, and license model. Access the service details pages for information on features, service plans, related tools, APIs, and documentation: [SAP Discovery Center: Service Catalog](#).

## **Neo Environment**

To access the documentation for the Neo environment, see [SAP BTP, Neo environment](#).

To learn more about why and how to migrate your scenarios in the Neo environment, see [Migrating from the SAP BTP Neo Environment to the Multi-Cloud Foundation](#).

## **All SAP BTP Resources**

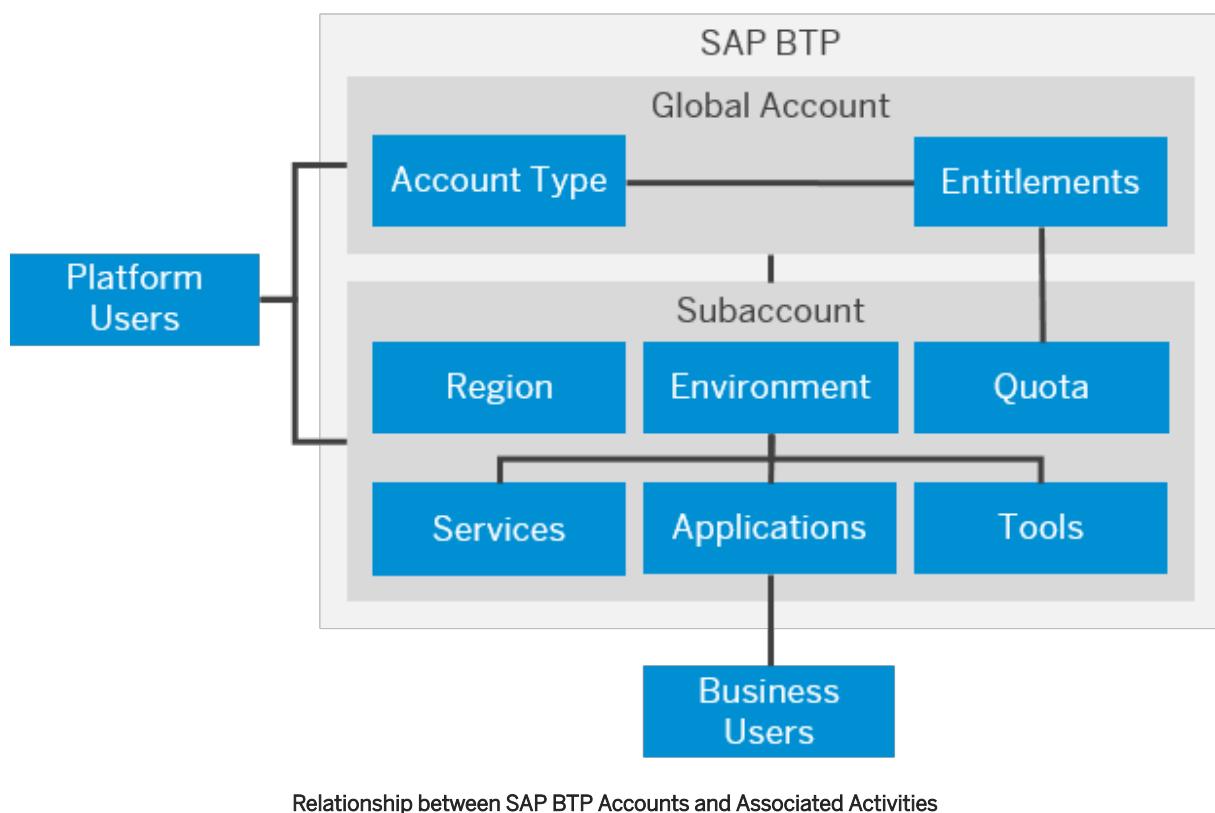
Access all of our resources on the product page: [SAP Business Technology Platform \(SAP BTP\)](#).

## 2 Basic Platform Concepts

SAP BTP offers users the ability to turn data into business value, compose end-to-end business processes, and build and extend SAP applications quickly.

SAP Business Technology Platform is built on a multi-cloud foundation, which lets you choose from different infrastructures and runtimes. The services and solutions of SAP BTP are available on multiple cloud infrastructure providers, and it supports different runtimes, such as Cloud Foundry, ABAP, and Kyma, as well as multiple different regions, and a broad choice of programming languages.

The central point of entry to the platform is the **SAP BTP cockpit**, where you can access your accounts and applications and manage all activities associated with them.



### Solutions and Services

SAP BTP offers fast in-memory processing, sustainable, agile **solutions** and **services** to integrate data and extend applications, and fully embedded analytics and intelligent technologies.

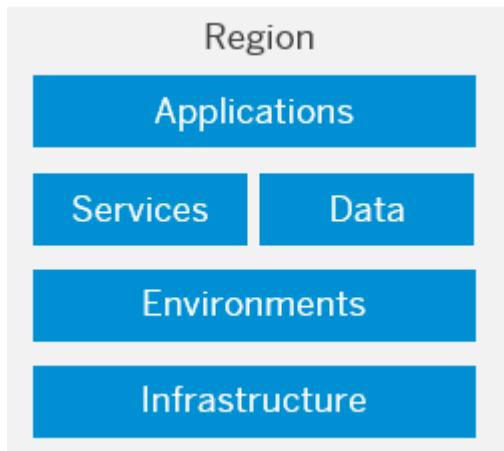
**Services** enable, facilitate, or accelerate the development of business applications and other platform services on SAP BTP.

For a complete list of services and capabilities, see [SAP Discovery Center: Services](#).

For more information, see [Solutions and Services \[page 16\]](#).

## Regions

You can deploy applications in different **regions**. Each region represents a geographical location (for example, Europe, US East) where applications, data, or services are hosted.



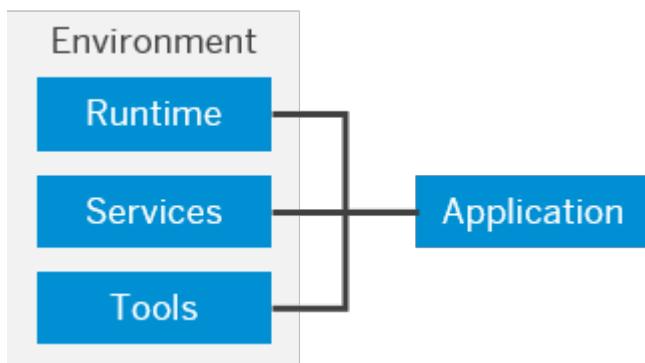
Regions are provided either by SAP or by our Infrastructure-as-a-Service (IaaS) partners Amazon Web Services (AWS), Microsoft Azure, Google Cloud, and Alibaba Cloud. The third-party region providers operate the infrastructure layer of the regions, whereas SAP operates the platform layer and Cloud Foundry.

A region is chosen at the subaccount level. For each subaccount, you select exactly one region (that is one data center).

For more information, see [Regions \[page 17\]](#).

## Environments

Environments constitute the actual platform-as-a-service offering of SAP BTP that allows for the development and administration of business applications. Environments are anchored in SAP BTP on subaccount level.



SAP BTP provides the following environments:

- [Cloud Foundry Environment \[page 49\]](#)

The Cloud Foundry environment enables you to develop new business applications and business services, supporting multiple runtimes, programming languages, libraries, and services. You can leverage a multitude of buildpacks, including community innovations and self-developed buildpacks.

- [ABAP Environment \[page 65\]](#)

Within the Cloud Foundry environment, you can create a new space for ABAP development. This is what we refer to as the ABAP environment. It allows you to create extensions for ABAP-based products, such as SAP S/4HANA Cloud, and develop new cloud applications. You can transform existing ABAP-based custom code or extensions to the cloud.

- [Kyma Environment \[page 70\]](#)

SAP BTP, Kyma runtime provides a fully managed cloud-native Kubernetes application runtime based on the open-source project "Kyma". Based on modular building blocks, Kyma runtime includes all the necessary capabilities to simplify the development and to run enterprise-grade cloud-native applications.

- [Neo Environment \[page 81\]](#)

The Neo environment lets you develop HTML5, Java, and SAP HANA extended application services (SAP HANA XS) applications. You can also use the UI Development Toolkit for HTML5 (SAPUI5) to develop rich user interfaces for modern web-based business applications.

→ Remember

SAP Business Technology Platform, Neo environment will sunset on **December 31, 2028**, subject to terms of customer or partner contracts.

For more information, see SAP Note [3351844](#).

## Enterprise and Trial Accounts

SAP BTP provides different types of global accounts, **enterprise** and **trial**. The type you choose determines pricing, conditions of use, resources, available services, and hosts.

Global Account

Trial Account

Global Account

Enterprise Account

- A **trial account** lets you try out the platform for free. Access is open to everyone. Trial accounts are intended for personal exploration, and not for production use or team development. They allow restricted use of the platform resources and services.

For more information, see [Trial Accounts and Free Tier \[page 82\]](#).

- An **enterprise account** is usually associated with one SAP customer or partner and contains their purchased entitlements to platform resources and services. It groups together different subaccounts that an administrator makes available to users for deploying applications.

For more information, see [Enterprise Accounts \[page 84\]](#).

## Commercial Models

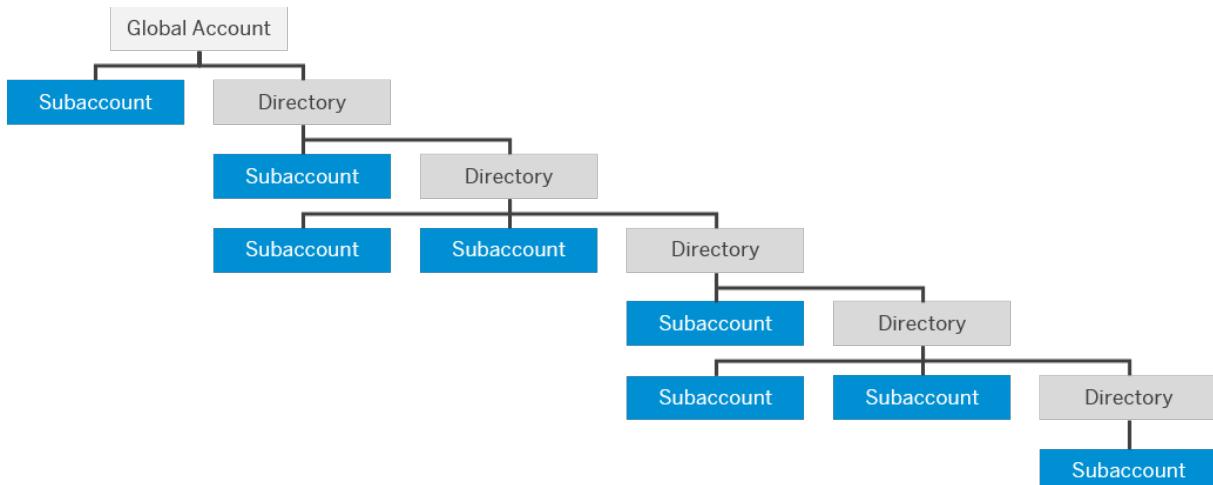
SAP BTP offers two different commercial models:



- **Consumption-based commercial model:** Your organization receives access to all current and future services that are eligible for this model. You have complete flexibility to turn services on and off and to switch between services as your business requires throughout the duration of your contract. This commercial model is available in the following flavors: SAP BTP Enterprise Agreement (SAP BTPEA), Cloud Platform Enterprise Agreement (CPEA), and Pay-As-You-Go for SAP BTP.  
For more information, see [What Is the Consumption-Based Commercial Model? \[page 87\]](#)
- **Subscription-based commercial model:** Your organization subscribes only to the services that you plan to use. You can then use these services at a fixed cost, irrespective of consumption.  
For more information, see [What Is the Subscription-Based Commercial Model? \[page 90\]](#)

## Account Model

The SAP BTP cockpit is structured according to global accounts, directories, and subaccounts:



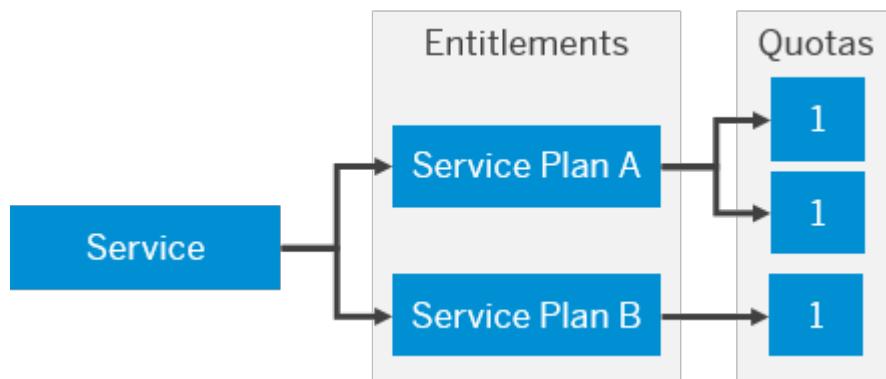
- A **global account** is the realization of a contract you or your company has made with SAP. A global account is used to manage subaccounts, members, entitlements and quotas. You receive entitlements and quotas to use platform resources per global account and then distribute the entitlements and quotas to the subaccount for actual consumption. There are two types of commercial models for global accounts: consumption-based model and subscription-based model. See [Commercial Models \[page 86\]](#)
- With **directories**, you can organize and manage your subaccounts according to your technical and business needs.

- **Subaccounts** let you structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements.
- In the SAP BTP, Cloud Foundry environment, the subaccount is divided into one or more spaces, which is where application development, deployment, and maintenance take place.

For more information, see [Account Model \[page 94\]](#).

## Entitlements and Quotas

When you purchase an enterprise account, you're entitled to use a specific set of resources, such as the amount of memory that can be allocated to your applications.



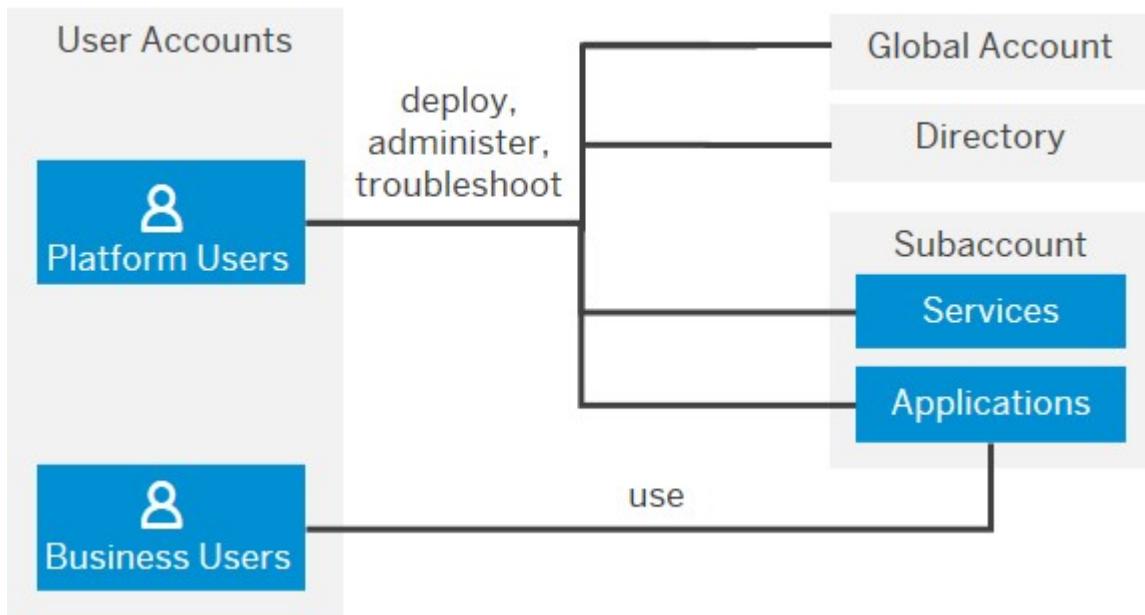
- On SAP BTP, all external dependencies such as databases, messaging systems, file systems, and so on, are **services**. In this context, multitenant applications and environments are considered services. Each service has one or more **service plans** available. A service plan is the representation of the costs and benefits for a given variant of a particular service. For instance, a database may be configured with various "T-shirt sizes", each of which is a different service plan.
- An **entitlement** is your right to provision and consume a resource. In other words, entitlements are **the service plans** that you're entitled to use.
- A **quota** represents the numeric quantity of a service plan that you're entitled to consume in your global account and its subaccounts.

For more information, see [Entitlements and Quotas \[page 100\]](#).

## User and Member Management

On SAP BTP, user management takes place at all levels from global account to environment. There are different types of users, such as depending on their roles in the company.

**User accounts** enable users to log on to SAP BTP, access subaccounts, and to use applications according to the permissions granted to them. We distinguish between two types of users:



- **Platform users** are usually developers, administrators or operators who deploy, administer, and troubleshoot accounts, applications and services on SAP BTP.
- **Business users** use the applications that are deployed to SAP BTP. For example, the end users of SaaS apps or services, such as SAP Build Work Zone, or end users of your custom applications are business users.

**Member management** refers to managing permissions for platform users. A member is a user who is assigned to an SAP BTP global account or subaccount. Administrators can add users to global accounts and subaccounts and assign roles to them as needed. You can use predefined roles, for example the administrator role for managing subaccount members.

**User management** refers to managing authentication and authorization for your business users.

For more information, see [User and Member Management \[page 104\]](#).

## Tools, Programming Models, Programming Languages, and APIs

SAP BTP provides various programming languages and tools for your development project:

- SAP BTP includes many tools to help you develop and manage applications, and connect them to your on-premise systems. For more information, see [Tools \[page 121\]](#).
- SAP BTP supports many different programming languages; the availability of each depends on the development environment you're using. For more information, see [Programming Languages \[page 123\]](#).
- The SAP Cloud Application Programming Model offers a consistent end-to-end programming model that includes languages, libraries, and APIs that are tailored for full-stack development on SAP BTP. For more information, see [Programming Models \[page 124\]](#).
- Depending on your use case, you can choose between different offerings for continuous integration and delivery. For more information, see [Continuous Integration and Delivery \(CI/CD\) \[page 125\]](#).
- Discover and consume APIs to manage, build, and extend the core capabilities of SAP BTP. For more information, see [APIs \[page 128\]](#).

## 2.1 Solutions and Services

Consume the solutions and services by SAP BTP according to your preferred development environment and use cases.

### Solutions

SAP BTP offers fast in-memory processing, sustainable, agile **solutions** and **services** to integrate data and extend applications, and fully embedded analytics and intelligent technologies.

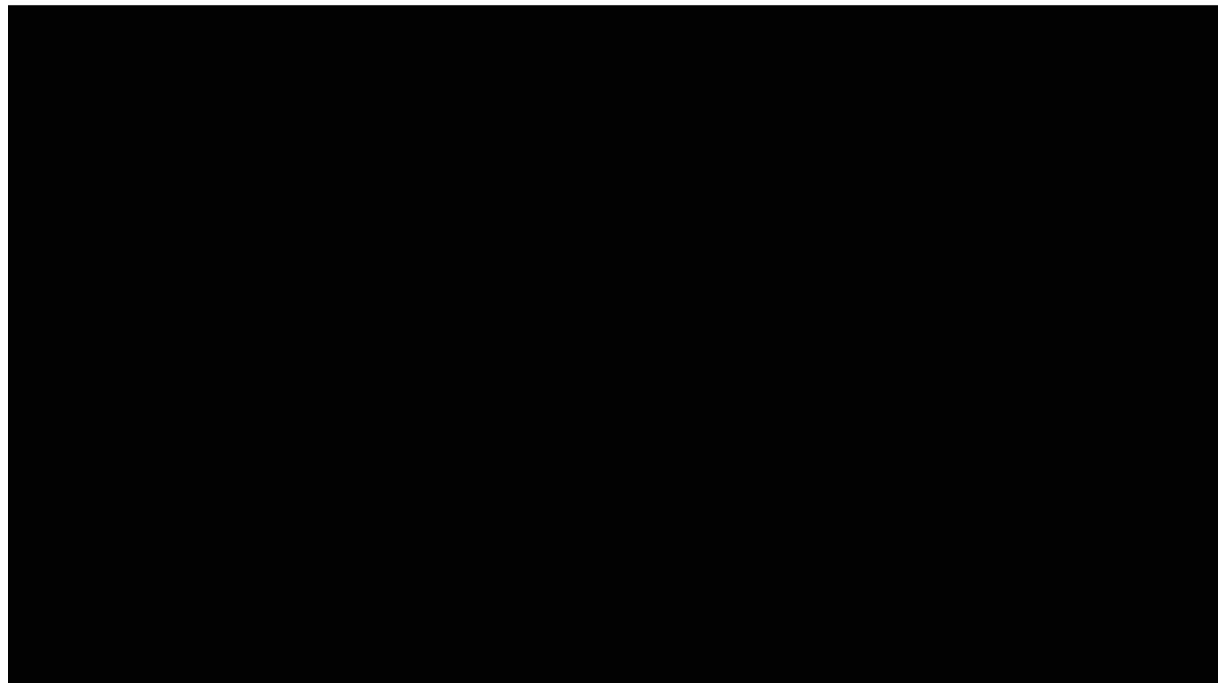
### Services

**Services** enable, facilitate, or accelerate the development of business applications and other platform services on SAP BTP. Services are grouped into the following service types:

- **Business services:** Services that enable, facilitate, or accelerate the development of business process components or provide industry-specific functionalities or content within a business application.
- **Technical services:** Services that enable, facilitate, or accelerate the development of general or domain independent content within a business application, independent of the application's business process or task.

You find all available services, solutions, and use cases in the [SAP Discovery Center](#).

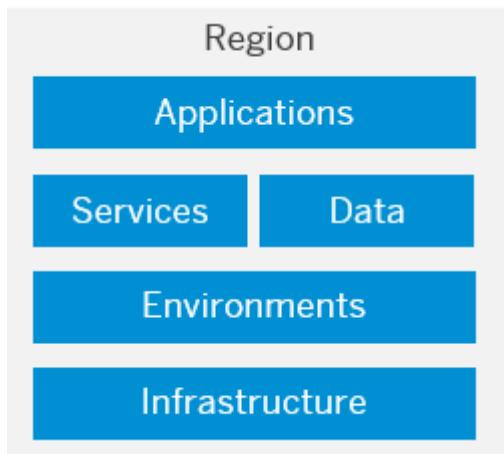
Use the service catalog to access service-specific resources:



## 2.2 Regions

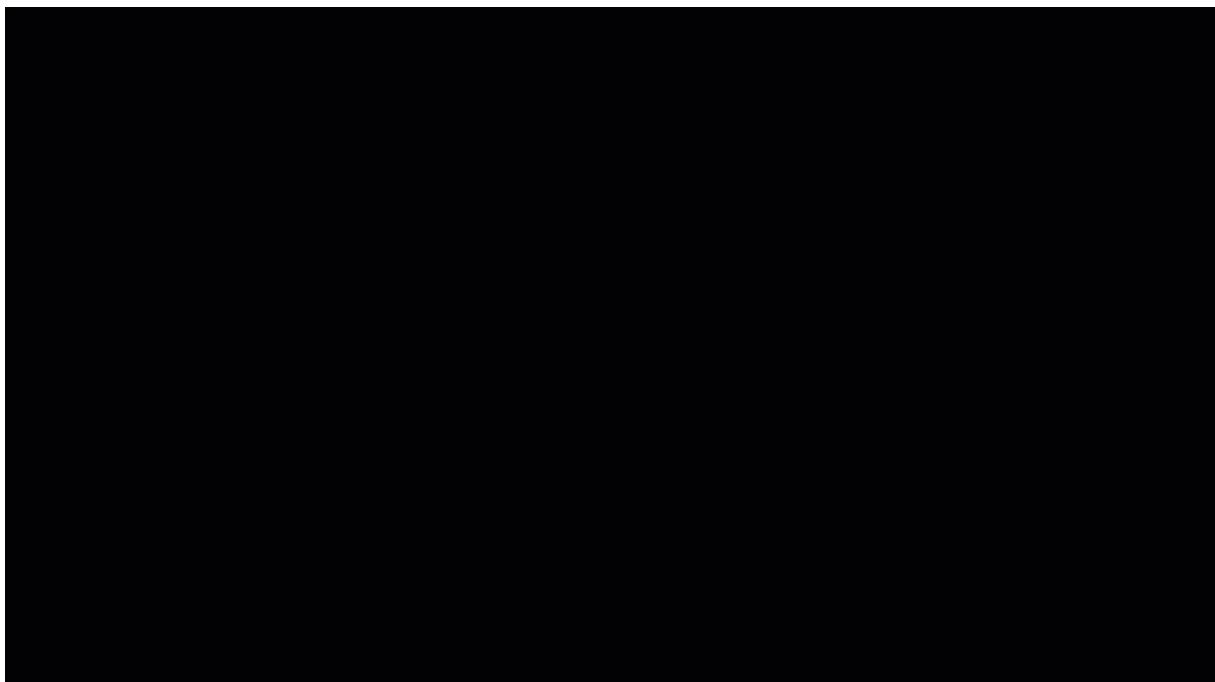
You can deploy applications in different **regions**. Each region represents a geographical location (for example, Europe, US East) where applications, data, or services are hosted.

### About SAP BTP Regions



Regions are provided either by SAP or by our Infrastructure-as-a-Service (IaaS) partners Amazon Web Services (AWS), Microsoft Azure, Google Cloud, and Alibaba Cloud. The third-party region providers operate the infrastructure layer of the regions, whereas SAP operates the platform layer and Cloud Foundry.

For an overview of all available regions, see [SAP Discovery Center](#):



## Selecting a Region

A region is chosen at the subaccount level. For each subaccount, you select exactly one region. The selection of a region is dependent on many factors: For example, application performance (response time, latency) can be optimized by selecting a region close to the user. For more information, see [Selecting a Region](#).

## Deploying Applications in Regions

When deploying applications, consider that a subaccount is associated with a particular region and that this is independent of your own location. You may be located in the United States, for example, but operate your subaccount in a region in Europe. For more information on subaccounts, see [Subaccounts \[page 95\]](#).

To deploy an application in more than one region, execute the deployment separately for each host. For more information, see [Deploy an Application \[page 2491\]](#).

Within a region, there can be multiple instances of the SAP BTP, Cloud Foundry environment. When creating a subaccount, SAP BTP automatically assigns the account to a specific instance of the environment. Several subaccounts of the same global account can be located in different datacenters in one region. This also affects the format of the API endpoint URL that is displayed in the cockpit after enabling Cloud Foundry in your subaccount. There are two possible formats for the API endpoint URL, either displayed with or without an index. Here's an example for **eu10**:

### Example

- <https://api.cf.eu10.hana.ondemand.com>
- <https://api.cf.eu10-<XXX>.hana.ondemand.com>

In both cases, the subaccount is located in the region eu10. The differences in the URLs are only an indicator of technical details on the side of SAP BTP and do not affect the functionality of your applications. For information on enabling Cloud Foundry, see [Create Orgs \[page 2435\]](#).

## High Availability

SAP has a number of processes in place to support resilience in SAP BTP, and provides different offerings so that you can support the high availability of your applications. For more information, see [Resilience, High Availability, and Disaster Recovery \[page 45\]](#).

## EU Access

Some customer contracts include EU Access, which restricts processing of personal data to EEA/Switzerland. If the global account is marked with EU Access, the actual EU Access compliance status of subaccounts will be displayed during creation of subaccounts.

### Note

If you require a subaccount with EU Access, make sure to select a provider and region where EU Access is available. Regions that allow for EU Access are labeled as such in the map view filter in the SAP Discovery Center.

For some services, EU Access is generally not available, not even if the provider and region support EU Access.

## Related Information

[Regions and API Endpoints Available for the Cloud Foundry Environment \[page 20\]](#)

[Regions and API Endpoints for the ABAP Environment \[page 37\]](#)

[Regions for the Kyma Environment \[page 41\]](#)

## 2.2.1 Regions and API Endpoints Available for the Cloud Foundry Environment

Regions for Enterprise Accounts

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon	
Microsoft Azure	eu20	Europe (Netherlands)	cf-eu20	West Europe	<b>cf-eu20:</b> 52.149.67.35, 20.82.96.175, 20.82.96.178, 20.82.96.211, 20.82.96.244, 20.82.96.220, 20.82.96.227, 20.82.97.50, 20.82.96.240, 20.82.96.234, 20.82.97.38, 20.82.96.222, 20.82.96.233, 20.82.96.248, 20.82.97.31, 20.82.97.45, 52.149.96.147, 20.56.169.152, 20.56.169.69, 20.56.169.0, 20.56.169.41, 20.56.169.58, 20.56.169.161, 20.56.169.116, 20.56.169.167, 20.56.169.50, 20.56.169.175, 20.56.169.131, 20.56.169.66, 20.56.169.71, 20.56.169.138, 20.56.169.91, 52.142.226.14, 20.86.1.84, 20.86.1.80, 20.86.0.233, 20.86.1.131, 20.86.1.54, 20.86.1.128, 20.86.1.134, 20.86.1.163, 20.86.1.15,	<b>cf-eu20:</b> 40.119.153.88 <b>cf-eu20-001:</b> 20.82.83.59	<b>cf-eu20:</b> <b>cf-eu20:</b> api.cf.eu 20.hana .on- .de- mand.co mand.co m m	<b>cf-</b> <b>eu20-0</b> <b>01:</b>	<b>cf-</b> <b>eu20-0</b> <b>01:</b>	<a href="#">Open Cockpit</a> 

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon	
					20.86.0.250, 20.86.1.107, 20.86.1.157, 20.86.0.253, 20.86.1.12, 20.86.1.97	<b>cf-eu20-001:</b>  20.54.248.90, 20.54.248.142, 20.54.250.88, 20.54.250.91, 20.56.17.109, 20.56.18.130, 20.56.18.132, 20.56.18.179, 20.86.17.114, 20.86.17.142, 20.86.17.153, 20.86.17.159				
Micro- soft Azure	ap20	Aus- tralia (Syd- ney)	cf-ap20	Aus- tralia East	40.82.211.52, 40.82.206.131, 20.70.176.247, 20.40.81.59, 20.40.80.246, 20.40.81.36, 20.70.208.178, 20.70.208.228, 20.70.208.235, 20.70.201.155, 20.70.201.89, 20.70.201.66	20.53.99.41	api.cf.ap 20.hana .onde- mand.co m	ap20.ha na.on- de- mand.co m	<a href="#">Open</a> <a href="#">Cockpit</a> 	
Micro- soft Azure	ap21	Singa- pore	cf-ap21	South- east Asia	40.90.179.153, 20.198.169.214, 20.198.168.45, 20.198.169.5, 40.90.170.226, 20.198.225.78, 20.198.225.102, 20.198.225.27, 40.90.162.117, 20.191.154.174, 20.191.154.191, 20.191.154.193	20.184.61.122	api.cf.ap 21.hana .onde- mand.co m	ap21.ha na.on- de- mand.co m	<a href="#">Open</a> <a href="#">Cockpit</a> 	

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Micro-soft Azure	br20	Brazil (São Paulo)	cf-br20	Brazil South	20.206.161.114, 20.206.249.230, 191.233.21.237	4.228.118.21	api.cf.br20.hana.on-demand.com	br20.hana.on-demand.com	<a href="#">Open Cockpit</a>
Micro-soft Azure	us20	US West (WA)	cf-us20	West US 2	40.90.195.191, 20.57.129.106, 20.57.128.95, 20.57.128.118, 40.90.209.71, 20.72.210.109, 20.72.209.240, 20.72.209.187, 40.90.200.224, 40.90.201.197, 40.90.201.85, 40.90.200.237	40.91.120.100	api.cf.us20.hana.on-demand.com	us20.hana.on-demand.com	<a href="#">Open Cockpit</a>
Micro-soft Azure	jp20	Japan (Tokyo)	cf-jp20	Japan East	52.185.186.130, 20.194.193.229, 20.194.193.167, 20.194.194.97, 40.81.200.207, 20.78.122.9, 20.78.121.237, 20.78.122.8, 20.40.96.175, 20.78.2.104, 20.78.2.106, 20.78.2.107	20.43.89.91	api.cf.jp20.hana.on-demand.com	jp20.hana.on-demand.com	<a href="#">Open Cockpit</a>
Micro-soft Azure	us21	US East (VA)	cf-us21	East US	40.90.251.147, 52.146.1.155, 20.51.255.236, 52.146.1.223, 40.90.232.167, 20.55.49.185, 20.55.49.92, 20.55.49.186, 40.90.231.101, 52.151.248.29, 52.146.15.82, 52.146.10.227	40.88.52.17	api.cf.us21.hana.on-demand.com	us21.hana.on-demand.com	<a href="#">Open Cockpit</a>

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon
Micro-soft Azure	ch20	Switzer-land (Zurich)	cf-ch20	Switzer-land North	20.208.128.83, 20.208.128.86, 20.208.128.87, 20.208.128.88, 51.103.208.79, 51.103.208.81, 51.103.208.85, 51.103.208.87, 51.107.2.38, 51.107.2.50, 51.107.2.52, 51.107.2.54	20.208.56.83	api.cf.ch 20.hana .onde- m	ch20.ha na.on- de- mand.co m	<a href="#">Open Cockpit</a> 
Amazon Web Services	br10	Brazil (São Paulo)	cf-br10	sa-east-1	52.67.245.111, 18.231.45.151, 54.207.173.126, 18.230.81.234, 18.229.169.29, 54.94.110.127, 18.231.101.158, 177.71.170.199, 54.232.227.140, 52.67.251.43, 54.232.20.181, 54.94.136.11, 52.67.221.224, 18.229.54.222, 54.232.250.83, 18.228.194.102, 18.228.198.142, 177.71.168.150	18.229.91.150, 52.67.135.4, 54.232.179.204, 18.228.53.198, 52.67.149.240, 54.94.179.209	api.cf.br 10.hana .onde- m	br10.ha na.on- de- mand.co m	<a href="#">Open Cockpit</a> 

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Amazon Web Services	jp10	Japan (Tokyo)	cf-jp10	ap-north-east-1	54.238.10.97, 54.250.43.250, 52.192.218.156, 35.74.54.33, 18.177.86.79, 35.74.144.49, 3.114.115.232, 18.179.150.168, 54.249.134.63, 18.179.66.68, 54.250.33.48, 54.95.22.24, 52.198.77.221, 35.73.255.50, 54.178.62.192, 3.113.232.224, 52.198.66.114, 13.230.215.218	3.114.248.68, 3.113.252.15, 13.114.117.83, 18.178.155.134, 57.180.140.5, 57.180.145.179	api.cf.jp 10.hana. onde- mand.co m	jp10.han a.onde- mand.co m	<a href="#">Open Cockpit</a>
Amazon Web Services	ap10	Australia (Sydney)	cf-ap10	ap-south-east-2	52.62.223.36, 13.55.100.204, 13.54.168.75, 13.55.239.117, 13.210.173.131, 13.54.77.205, 13.237.182.31, 52.65.102.82, 54.79.72.145, 13.236.142.207, 54.79.43.227, 13.54.252.220, 54.79.26.135, 13.54.220.129, 13.236.59.235, 13.238.9.23, 54.153.242.51, 3.105.234.54	13.236.220.84, 13.211.73.244, 3.105.95.184, 13.55.188.95, 3.105.212.249, 3.106.45.106	api.cf.ap 10.hana. na.on- onde- mand.co m m	ap10.ha na.on- de- mand.co m m	<a href="#">Open Cockpit</a>

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Amazon Web Services	ap11	Asia Pacific (Singapore)	cf-ap11	ap-south-east-1	13.251.40.148, 13.228.68.14, 13.251.49.36, 52.76.185.92, 13.229.13.240, 54.251.74.134, 52.220.111.202, 54.179.77.154, 52.76.123.164, 54.179.253.138, 13.213.119.83, 3.1.38.48, 52.76.114.209, 13.213.105.43, 13.213.132.88, 13.250.92.77, 18.140.150.56, 18.140.255.164	18.139.147.53, 3.0.9.102, 18.140.39.70, 13.229.158.122, 18.140.228.217, 52.74.215.89	api.cf.ap11.hana.on-demand.co	ap11.hanona.on-demand.co	<a href="#">Open Cockpit</a>
Amazon Web Services	ap12	Asia Pacific (Seoul)	cf-ap12	ap-north-east-2	15.165.116.197, 54.180.53.68, 3.35.252.222, 52.78.49.16, 15.164.33.162, 3.34.19.116, 3.36.2.67, 3.35.57.231, 15.164.254.80, 3.36.165.189, 52.78.38.74, 15.165.249.251, 13.124.251.247, 13.124.16.17, 15.165.83.237	3.35.106.215, 3.35.255.45, 3.35.215.12, 13.209.236.215, 43.201.194.105, 43.202.204.5	api.cf.ap12.hana.on-demand.co	ap12.hanona.on-demand.co	<a href="#">Open Cockpit</a>

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Amazon Web Services	ca10	Canada (Montreal)	cf-ca10	ca-central-1	35.182.118.205, 35.182.198.31, 3.98.159.3, 3.96.101.45, 15.222.120.34, 3.96.14.215, 3.97.48.154, 3.97.119.250, 35.182.95.49, 3.97.228.23, 35.182.185.156, 3.98.252.245, 15.223.62.0, 99.79.181.241, 3.98.167.60, 99.79.110.245, 52.60.239.204, 52.60.212.33	35.183.74.34, 35.182.75.101, 3.98.102.153, 15.157.88.166, 3.98.202.222, 52.60.210.33	api.cf.ca 10.hana. onde- m	ca10.ha na.on- de- mand.co m	Feature Set B 

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon
Amazon Web Services	eu10	Europe (Frankfurt)	cf-eu10	eu-central-1	<p>cf-eu10:</p> <p>52.59.128.222,</p> <p>52.28.241.88,</p> <p>18.184.81.94,</p> <p>3.67.200.70,</p> <p>3.68.51.135,</p> <p>3.124.174.204,</p> <p>3.68.31.37,</p> <p>3.67.58.183,</p> <p>3.67.0.172,</p> <p>3.67.244.62,</p> <p>3.126.117.58,</p> <p>3.66.100.105,</p> <p>3.68.13.226,</p> <p>3.126.45.133,</p> <p>3.67.249.135,</p> <p>18.194.183.183,</p> <p>3.67.246.74,</p> <p>3.66.68.201,</p> <p>3.68.0.70,</p> <p>52.28.56.202,</p> <p>3.126.95.250,</p> <p>3.66.68.127,</p> <p>18.195.244.40,</p> <p>3.67.107.121,</p> <p>3.67.24.253,</p> <p>18.193.50.255,</p> <p>3.121.35.143</p>	<p>cf-eu10:</p> <p>3.124.208.223,</p> <p>3.122.209.241,</p> <p>3.124.222.77,</p> <p>18.159.31.22,</p> <p>3.69.186.98,</p> <p>3.77.195.119</p> <p>cf-eu10-002:</p> <p>3.126.229.22,</p> <p>18.193.180.19,</p> <p>3.64.227.236,</p> <p>18.153.123.11,</p> <p>3.121.37.195,</p> <p>3.73.215.90</p> <p>cf-eu10-003:</p> <p>3.127.77.3,</p> <p>3.64.196.58,</p> <p>18.156.151.247,</p> <p>18.197.252.154,</p> <p>3.79.137.29,</p> <p>52.58.93.50</p> <p>cf-eu10-004:</p> <p>3.70.38.218,</p> <p>18.196.206.8,</p> <p>3.65.185.47,</p> <p>3.73.109.100,</p> <p>3.73.8.210,</p> <p>52.59.18.183</p> <p>cf-eu10-005:</p> <p>3.78.172.245,</p> <p>3.78.172.245,</p> <p>3.122.31.132,</p> <p>3.122.31.132,</p> <p>18.193.56.244,</p> <p>18.193.56.244</p> <p>cf-eu10-003:</p> <p>3.64.131.199,</p> <p>3.64.88.217,</p>	<p>cf-eu10:</p> <p>api.cf.eu</p> <p>eu10.hana.</p> <p>na.on-</p> <p>onde-</p> <p>de-</p> <p>mand.co</p> <p>m</p> <p>cf-</p> <p>eu10-0</p> <p>02:</p> <p>eu10-0</p> <p>02:</p> <p>api.cf.eu</p> <p>eu10-00</p> <p>10-002.</p> <p>2.hana.o</p> <p>hana.on</p> <p>nde-</p> <p>de-</p> <p>mand.co</p> <p>m</p> <p>cf-</p> <p>eu10-0</p> <p>03:</p> <p>eu10-0</p> <p>03:</p> <p>api.cf.eu</p> <p>3.hana.o</p> <p>10-003.</p> <p>nde-</p> <p>hana.on</p> <p>mand.co</p> <p>m</p> <p>mand.co</p> <p>m</p> <p>cf-</p> <p>eu10-0</p> <p>04:</p> <p>eu10-0</p> <p>04:</p> <p>eu10-0</p> <p>eu10-00</p> <p>10-004.</p> <p>mand.co</p> <p>hana.on</p> <p>m</p> <p>cf-</p> <p>eu10-0</p> <p>05:</p> <p>eu10-0</p> <p>eu10-00</p> <p>5.hana.o</p> <p>nde-</p> <p>api.cf.eu</p> <p>mand.co</p> <p>10-005.</p> <p>m</p> <p>hana.on</p>	<p>cf-eu10:</p> <p>Open Cockpit</p> 	

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon	
					3.64.142.243, 18.198.18.157, 3.68.40.83, 3.67.235.98, 3.68.17.221, 18.198.149.19, 3.68.38.23		de-demand.com			
					<b>cf-eu10-004:</b>  3.69.195.103, 3.64.170.167, 3.68.176.248, 3.121.49.211, 18.197.219.60, 3.70.38.84					
					<b>cf-eu10-005:</b>  35.159.192.144/28					
Amazon Web Services	eu11	Europe (Frankfurt)	cf-eu11	eu-central-1	18.156.140.38, 3.121.55.100, 35.156.198.246, 52.59.77.121, 18.185.57.85, 3.121.79.209, 3.67.237.8, 35.156.31.32, 3.65.63.251, 3.122.176.63, 18.198.13.57, 18.157.114.142, 18.184.172.97, 18.159.180.188, 35.157.5.44	18.156.209.198, 18.157.105.117, 3.124.207.41, 3.66.26.249, 3.72.216.204, 3.74.99.245	api.cf.eu11.hana.ondemand.com	eu11.hana.a.ondemand.com	<a href="#">Open Cockpit</a>	

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	<p><b>cf-us10:</b></p> <p>52.200.16.71, 52.23.123.125, 52.202.170.155, 18.210.47.160, 18.211.235.11, 54.156.172.106, 34.234.191.59, 34.192.134.47, 18.204.173.15, 3.213.197.54, 184.73.43.82, 52.20.242.182, 3.216.16.207, 34.225.190.250, 52.2.110.230, 54.234.93.200, 35.153.88.132, 52.204.111.138, 3.88.250.160, 54.221.30.91, 52.71.83.110, 52.200.165.163, 54.208.119.130, 34.202.136.35, 34.192.100.96, 54.85.65.82, 54.205.71.200</p> <p><b>cf-us10-001:</b></p> <p>52.0.214.195, 18.213.153.162, 54.227.144.195, 3.225.73.158, 34.233.151.91, 23.23.172.117, 3.222.22.16, 34.238.1.234, 34.194.239.31, 3.225.44.56, 34.201.208.150, 75.101.157.228</p>	<p><b>cf-us10:</b></p> <p>52.4.101.240, 52.23.1.211, 52.23.189.23, 18.213.242.208, 3.214.110.153, 34.205.56.51 <b>cf-us10-001:</b> 3.227.182.44, 52.86.131.53, 3.220.114.17, 44.218.82.203, 44.219.57.163, 50.16.106.103 <b>cf-us10-002:</b> 107.20.66.86, 54.234.152.59, 34.202.68.0, 3.214.116.95, 54.144.230.36, 54.226.37.104 <b>cf-us10-001:</b> 52.0.214.195, 18.213.153.162, 54.227.144.195, 3.225.73.158, 34.233.151.91, 23.23.172.117, 3.222.22.16, 34.238.1.234, 34.194.239.31, 3.225.44.56, 34.201.208.150, 75.101.157.228</p>	<p><b>cf-us10:</b></p> <p>api.cf.us 10.hana. onde- mand.co m</p> <p><b>cf- us10-00 1:</b></p> <p>api.cf.us 10-001. hana.on nde- mand.co m</p> <p><b>cf- us10-00 2:</b></p> <p>api.cf.us 10-002. hana.on nde- mand.co m</p>	<p><b>cf-us10:</b></p> <p>us10.ha na.on- de- mand.co m</p> <p><b>cf- us10-00 1:</b></p> <p>us10-00 1.hana.o</p> <p><b>cf- us10-00 2:</b></p> <p>us10-00 2.hana.o</p>	

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon	
<b>cf-us10-002:</b>										
Google Cloud	us30	US Cen- tral (IA)	cf-us30	us-cen- tral1	54.162.233.194, 34.206.160.141, 35.168.80.144, 18.232.28.65, 54.82.224.146, 3.221.4.74, 18.211.12.227, 54.159.45.198, 72.44.51.245	35.202.96.192, 35.193.171.152, 35.193.168.31, 35.202.69.204, 35.202.175.147, 35.193.69.164, 35.202.1.6, 23.236.63.113, 35.193.30.116, 35.202.66.196, 34.68.152.205, 35.222.158.222, 104.197.20.168, 35.232.105.70, 35.224.211.196, 35.222.192.158, 35.193.8.172, 34.171.4.220, 34.172.37.175, 34.170.206.220, 34.172.145.231, 35.222.38.254, 35.239.28.216, 34.134.91.47, 34.123.17.36, 35.202.205.85, 34.118.207.84, 35.193.6.192, 34.122.222.203, 104.197.157.121, 34.135.159.154, 35.223.208.27, 146.148.74.171, 34.132.192.46, 34.68.109.37, 104.198.49.58, 35.225.164.132	35.184.169.79	api.cf.us 30.hana .onde- m	us30.ha na.on- de- mand.co m	<a href="#">Open</a> <a href="#">Cockpit</a> 

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Pro- vider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming re- quests)	API End- point	Domain	Cockpit Logon
Google Cloud	eu30	Europe (Frankfurt)	cf-eu30	europe-west3	34.107.28.38, 34.141.10.217, 34.141.116.52, 34.141.1.228, 34.141.123.52, 34.141.125.107, 34.141.46.51, 34.89.130.182, 34.89.146.167, 34.89.203.91, 34.89.232.158, 34.89.243.40, 35.198.83.71, 35.234.65.38, 35.242.208.222, 35.246.155.42, 35.246.171.35, 34.141.28.26, 34.159.160.86, 34.107.19.175, 34.159.165.29, 35.242.240.154, 34.141.73.130, 34.159.27.236, 34.89.152.211, 35.242.194.75, 35.246.235.253, 34.159.127.190, 34.141.82.126, 35.234.69.102, 34.89.231.53, 34.159.188.133, 35.246.203.194, 34.159.201.78, 34.141.112.232, 35.198.84.213, 34.89.165.33	35.198.143.110	api.cf.eu 30.hana .onde- m	eu30.ha na.on- de- mand.co m	<a href="#">Open Cockpit</a> 

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Google Cloud	in30	India (Mumbai)	cf-in30	asia-south1	34.93.27.36, 34.93.89.145, 34.93.92.210, 34.93.137.163, 34.93.148.247, 34.93.155.252, 34.93.166.164, 34.93.180.0, 34.93.221.129, 35.200.131.125, 35.200.144.1, 35.200.175.62, 35.200.183.224, 35.200.194.175, 35.200.198.26, 35.200.209.142, 35.244.29.120, 35.200.137.225, 34.100.186.241, 35.200.169.254, 35.200.151.131, 35.200.252.103, 35.244.15.103, 35.244.16.76, 34.93.255.115, 35.244.53.153, 35.200.168.60, 35.200.222.30, 34.100.178.164, 35.244.2.193, 34.93.11.49, 34.100.211.195, 34.100.151.15, 34.93.95.83, 34.100.215.143, 34.93.205.174, 34.93.159.24	34.93.125.74	api.cf.in 30.hana .onde-m	in30.ha na.on-de mand.co m	<a href="#">Open Cockpit</a> 

IaaS Provider	Region	Region Name	Techni- cal Key	Techni- cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Google Cloud	il30	Israel (Tel Aviv)	cf-il30	me-west1	34.165.0.14, 34.165.0.115, 34.165.5.181, 34.165.5.246, 34.165.7.73, 34.165.12.173, 34.165.16.177, 34.165.16.210, 34.165.17.27, 34.165.18.240, 34.165.21.242, 34.165.24.112, 34.165.26.162, 34.165.37.171, 34.165.38.114, 34.165.40.240, 34.165.41.254, 34.165.80.207, 34.165.81.54, 34.165.110.15, 34.165.136.9, 34.165.150.108, 34.165.168.74, 34.165.171.197, 34.165.172.4, 34.165.194.20, 34.165.222.104, 34.165.223.90, 34.165.228.95, 34.165.231.165	34.165.59.26	api.cf.il30.hana.on-demand.com	il30.hana.on-demand.com	<a href="#">Open Cockpit</a>

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Google Cloud	sa30	KSA (Dam-mam)	cf-sa30	me-cen-tral2	34.166.4.164, 34.166.4.182, 34.166.8.22, 34.166.10.68, 34.166.19.145, 34.166.20.65, 34.166.37.166, 34.166.38.149, 34.166.39.3, 34.166.40.145, 34.166.41.104, 34.166.44.55, 34.166.46.73, 34.166.46.216, 34.166.47.18, 34.166.50.229, 34.166.52.69, 34.166.53.147, 34.166.55.78, 34.166.57.177, 34.166.60.109, 34.166.61.61, 34.166.61.119, 34.166.61.169, 34.166.61.173, 34.166.62.51, 34.166.62.206, 34.166.63.160, 34.166.64.6, 34.166.65.231	34.166.32.46	api.cf.sa30.hana.on-demand.com	sa30.hana.on-demand.com	<a href="#">Open Cockpit</a>

IaaS Provider	Region	Region Name	Techni-cal Key	Techni-cal Key of IaaS Provider	NAT IPs (egress, IPs for requests from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API End-point	Domain	Cockpit Logon
Google Cloud	ap30	Aus-tralia (Sydney)	cf-ap30	aus-tralia-south-east1	34.40.139.113, 34.40.148.106, 34.40.148.145, 34.87.197.203, 34.87.227.150, 34.87.246.15, 34.87.246.56, 34.116.64.18, 34.116.71.248, 34.116.93.109, 34.116.96.225, 34.116.98.18, 34.151.82.192, 34.151.84.167, 34.151.92.254, 34.151.95.105, 34.151.110.186, 34.151.113.55, 35.189.5.90, 35.189.6.201, 35.189.24.237, 35.189.29.223, 35.197.168.71, 35.201.9.202, 35.201.25.163, 35.201.28.91, 35.244.87.121, 35.244.96.216, 35.244.100.190, 35.244.124.253	35.244.71.16	api.cf.ap30.hana.on-demand.com	ap30.hana.on-demand.com	<a href="#">Open Cockpit</a> 
Alibaba Cloud	cn40	China (Shang-hai)	cf-cn40	cn-shang-hai	101.132.190.155, 106.14.165.33, 106.14.184.113	139.224.7.71	api.cf.cn40.platform.form.sa.pcloud.cn	cn40.platform.form.sa.pcloud.cn	<a href="#">Open Cockpit</a> 

## Regions for Trial Accounts

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	Trial NAT IPs (egress, IPs for requests)	from a Cloud Foundry app)	LB IPs (ingress, for incoming requests)	API Endpoint	Domain	Cockpit Logon
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	3.218.99.1 54, 52.72.147. 227, 3.218.112. 63	52.23.189. 23, 52.4.101.2 40, 52.23.1.21 1	api.cf.us10 .hana.on- de- mand.com	us10.hana. ondemand.com	Trial	
Microsoft Azure	ap21	Singapore	cf-ap21	Southeast Asia	52.139.216. .172, 20.195.24. 178, 20.195.9.1 69	20.184.61. 122	api.cf.ap21 .hana.on- de- mand.com	ap21.hana .ondemand.com	Trial	

### ⓘ Note

Trial accounts and subaccounts on trial can no longer be created on eu10, Europe (Frankfurt).

Existing trial accounts and subaccounts are not affected.

### ⓘ Note

In the Cloud Foundry environment, IPs are controlled by the respective IaaS provider (AWS, Azure, or Google Cloud). IPs may change due to network updates on the provider side. Any planned changes will be announced at least four weeks before they take effect.

### ⓘ Note

In the Cloud Foundry environment, the region in which a global account was created determines the API endpoint of all subaccounts associated with it. For example, subaccounts created in a global account in region **eu10** share the API endpoint URL `api.cf.eu10.hana.ondemand.com`.

## 2.2.2 Regions and API Endpoints for the ABAP Environment

Regions for Enterprise Accounts

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, IPs for requests from an ABAP System)	API Endpoint	Domain	Cockpit Logon
Amazon Web Services	ap10	Australia (Sydney)	cf-ap10	ap-south-east-2	54.153.226.137, 54.153.194.85, 54.79.209.86, 13.238.93.75	api.cf.ap10.hana.on-demand.com	ap10.hana.on-demand.com	<a href="https://apac cockpitpit.btp.cloud.sap/">https://apac cockpitpit.btp.cloud.sap/</a>
Amazon Web Services	br10	Brazil (Sao Paulo)	cf-br10	sa-east-1	52.67.140.201, 54.207.133.145, 15.229.97.244, 18.228.92.201	api.cf.br10.hana.on-demand.com	br10.hana.on-demand.com	<a href="https://amer cockpitpit.btp.cloud.sap/">https://amer cockpitpit.btp.cloud.sap/</a>
Amazon Web Services	ca10	Canada (Montreal)	cf-ca10	ca-central-1	15.222.180.159, 15.222.175.12, 52.60.183.108, 3.97.94.144	api.cf.ca10.hana.on-demand.com	ca10.hana.on-demand.com	<a href="https://amer cockpitpit.btp.cloud.sap/">https://amer cockpitpit.btp.cloud.sap/</a>

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, IPs for requests from an ABAP System)	API Endpoint	Domain	Cockpit Logon
Amazon Web Services	eu10	Europe (Frankfurt)	cf-eu10	eu-central-1	18.197.217.237, 18.198.153.44, 18.157.206.182, 52.57.94.154, 3.74.95.163, 18.156.20.40, 3.70.85.193, 3.76.177.92, 18.196.196.117, 3.127.41.81, 3.69.221.68, 3.67.230.143, 3.74.106.119, 18.199.199.153, 3.78.46.180	cf-eu10:api.cf.eu10.eu10.hana.hana.on-de-mand.comcf-eu10-002:eu10-002.api.cf.eu10.hana.on-002.hana.de-onde-mand.comcf-eu10-003:eu10-003.api.cf.eu10.hana.on-003.hana.de-onde-mand.comcf-eu10-004:eu10-004.api.cf.eu10.hana.on-004.hana.de-onde-mand.command.com	cf-eu10:eu10-002:eu10-002.api.cf.eu10.hana.on-002.hana.de-onde-mand.comcf-eu10-003:eu10-003.api.cf.eu10.hana.on-003.hana.de-onde-mand.comcf-eu10-004:eu10-004.api.cf.eu10.hana.on-004.hana.de-onde-mand.command.com	cf-eu10:emea cockpit.btp.cloud.sap/
Amazon Web Services	eu11	Europe (Frankfurt) EU Access	cf-eu11	eu-central-1	18.157.200.44, 3.121.238.156, 3.67.47.252, 18.195.136.83, 3.120.9.225, 3.68.253.186	api.cf.eu11.eu11.hana.hana.on-de-mand.com	eu11.hana.onde-mand.com	https://eu-access.cockpit.btp.cloud.sap/
Amazon Web Services	jp10	Japan (Tokyo)	cf-jp10	ap-north-east-1	35.75.28.56, 35.74.196.78, 35.74.158.17, 54.250.128.197, 52.196.122.86	api.cf.jp10.hana.on-de-mand.com	jp10.hana.onde-mand.com	https://apac.cockpit.btp.cloud.sap/
Amazon Web Services	ap11	Singapore	cf-ap11	ap-south-east-1	54.179.221.168, 18.140.151.124, 54.254.21.208, 54.251.135.238	api.cf.ap11.hana.on-de-mand.com	ap11.hana.onde-mand.com	https://apac.cockpit.btp.cloud.sap/

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, IPs for requests from an ABAP System)	API Endpoint	Domain	Cockpit Logon
Amazon Web Services	ap12	South Korea (Seoul)	cf-ap12	ap-north-east-2	3.39.57.235, 13.125.0.129, 13.124.63.148, 13.209.63.123	api.cf.ap12	ap12.hana.on-demand.com	<a href="https://apac cockpit.btp.cloud.sap/">https://apac cockpit.btp.cloud.sap/</a>
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	54.243.29.110, 18.215.92.120, 34.232.200.153, 18.232.247.104, 3.209.189.244, 18.204.158.200, 23.20.221.103, 54.86.32.250, 52.1.255.25	cf-us10:api.cf.us10	cf-us10:us10.hana.on-demand.com	<a href="https://amer cockpit.btp.cloud.sap/">https://amer cockpit.btp.cloud.sap/</a>
Microsoft Azure	eu20	Europe (Netherlands)	cf-eu20	westeurope	74.234.204.238, 98.71.236.50, 108.143.241.97	cf-eu20:api.cf.eu20	cf-eu20:eu20.hana.on-demand.com	<a href="https://emea cockpit.btp.cloud.sap/">https://emea cockpit.btp.cloud.sap/</a>
Microsoft Azure	ch20	Switzerland (Zurich) Azure EU Access	cf-ch20	switzerlandnorth	172.162.240.234, 51.107.40.177, 51.103.221.23	api.cf.ch20	ch20.hana.on-demand.com	<a href="https://eu-access cockpit.btp.cloud.sap/">https://eu-access cockpit.btp.cloud.sap/</a>

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, IPs for requests from an ABAP System)	API Endpoint	Domain	Cockpit Logon
Microsoft Azure	us20	US West (WA)	cf-us20	westus2	20.9.136.184, 172.179.0.166, 20.9.147.50	api.cf.us20.hana.on-demand.com	us20.hana.on-demand.com	<a href="https://amer cockpit.btp.cloud.sap/">https://amer cockpit.btp.cloud.sap/</a> 
Microsoft Azure	us21	US East (VA)	cf-us21	eastus	172.190.88.27, 20.83.171.94, 74.235.125.210	api.cf.us21.hana.on-demand.com	us21.hana.on-demand.com	<a href="https://amer cockpit.btp.cloud.sap/">https://amer cockpit.btp.cloud.sap/</a> 
Microsoft Azure	ap20	Australia (Sydney)	cf-ap20	australiaeast	20.211.81.46, 20.70.213.77, 20.211.46.196	api.cf.ap20.hana.on-demand.com	ap20.hana.on-demand.com	<a href="https://apac cockpit.btp.cloud.sap/">https://apac cockpit.btp.cloud.sap/</a> 
Microsoft Azure	ap21	Singapore	cf-ap21	southeastasia	20.6.9.203, 20.198.169.36, 20.195.25.14	api.cf.ap21.hana.on-demand.com	ap21.hana.on-demand.com	<a href="https://apac cockpit.btp.cloud.sap/">https://apac cockpit.btp.cloud.sap/</a> 
Microsoft Azure	jp20	Japan (Tokyo)	cf-jp20	japaneast	20.78.124.111, 20.63.141.36, 20.78.37.7	api.cf.jp20.hana.on-demand.com	jp20.hana.on-demand.com	<a href="https://apac cockpit.btp.cloud.sap/">https://apac cockpit.btp.cloud.sap/</a> 
Google Cloud	eu30	Europe (Frankfurt)	cf-eu30	europe-west3	34.141.88.79, 34.159.65.75, 34.141.107.207, 34.141.42.177, 34.141.126.80, 34.159.215.19	api.cf.eu30.hana.on-demand.com	eu30.hana.on-demand.com	<a href="https://emea cockpit.btp.cloud.sap/">https://emea cockpit.btp.cloud.sap/</a> 
Google Cloud	in30	India (Mumbai)	cf-in30	asia-south1	34.93.58.135, 35.200.177.49, 35.244.63.61, 34.93.41.126, 34.93.191.130, 35.200.168.232	api.cf.in30.hana.on-demand.com	in30.hana.on-demand.com	<a href="https://apac cockpit.btp.cloud.sap/">https://apac cockpit.btp.cloud.sap/</a> 

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	NAT IPs (egress, IPs for requests from an ABAP System)	API Endpoint	Domain	Cockpit Logon
Google Cloud	us30	US Central (IA)	cf-us30	us-central1	34.121.78.84, 34.122.132.185, 104.154.245.19, 34.171.92.122, 35.224.140.73, 34.31.173.8	api.cf.us30.hana.on-demand.com	us30.hana.on-demand.com	<a href="https://amer cockpit.btp.sap/">https://amer cockpit.btp.sap/</a>

Regions for Trial Accounts

IaaS Provider	Region	Region Name	Technical Key	Technical Key of IaaS Provider	Trial NAT IPs (egress, IPs for requests from a Cloud Foundry app)	API Endpoint	Domain	Cockpit Logon
Amazon Web Services	us10	US East (VA)	cf-us10	us-east-1	3.218.99.154, 52.72.147.227, 3.218.112.63	api.cf.us10.hana.on-demand.com	us10.hana.on-demand.com	Trial
Microsoft Azure	ap21	Singapore	cf-ap21	southeastasia	20.6.9.203, 20.198.169.36, 20.195.25.14	api.cf.ap21.hana.on-demand.com	ap21.hana.on-demand.com	Trial

## 2.2.3 Regions for the Kyma Environment

To work with the Kyma environment, you need to specify the region for both your subaccount and the cluster.

### ⓘ Note

In the Kyma environment, IP addresses for NAT Gateway that handles the egress traffic are configured dynamically. This means that you cannot identify the IP address of NAT Gateway in advance. However, once the IP address is assigned, it remains unchanged throughout the cluster's lifecycle.

## Subaccount Regions

The table lists the regions you can choose from when creating a subaccount.

## Subaccount Regions for Kyma

Global Account					
Type	Region	IaaS Provider	Technical Key	Region Name	Plan ID
Enterprise account	ap21	Microsoft Azure	cf-ap21	Singapore	azure
Partner Test, Demo, and Development account					azure_lite
Trial account					trial
Enterprise account	us20	Microsoft Azure	cf-us20	US West (WA)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	jp20	Microsoft Azure	cf-jp20	Japan (Tokyo)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	us21	Microsoft Azure	cf-us21	US East (VA)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	eu20	Microsoft Azure	cf-eu20	Europe (Netherlands)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	ap20	Microsoft Azure	cf-ap20	Australia (Sydney)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	br20	Microsoft Azure	cf-br20	Brazil (São Paulo)	azure
Partner Test, Demo, and Development account					azure_lite
Enterprise account	ch20	Microsoft Azure	cf-ch20	Switzerland (Zurich) EU Access	azure
Enterprise account	us10	Amazon Web Services	cf-us10	US East (VA)	aws
Trial account					trial

### Global Account

Type	Region	IaaS Provider	Technical Key	Region Name	Plan ID
Enterprise account	eu10	Amazon Web Services	cf-eu10	Europe (Frankfurt)	aws
Trial account					trial
Enterprise account	eu11	Amazon Web Services	cf-eu11	Europe (Frankfurt) EU Access	aws
Enterprise account	br10	Amazon Web Services	cf-br10	Brazil (São Paulo)	aws
Enterprise account	jp10	Amazon Web Services	cf-jp10	Japan (Tokyo)	aws
Enterprise account	ca10	Amazon Web Services	cf-ca10	Canada (Montreal)	aws
Enterprise account	ap12	Amazon Web Services	cf-ap12	South Korea (Seoul)	aws
Enterprise account	ap10	Amazon Web Services	cf-ap10	Australia (Sydney)	aws
Enterprise account	ap11	Amazon Web Services	cf-ap11	Singapore	aws
Enterprise account	us30	Google Cloud	cf-us30	US Central (IA)	gcp
Enterprise account	eu30	Google Cloud	cf-eu30	Europe (Frankfurt)	gcp
Enterprise account	in30	Google Cloud	cf-in30	India (Mumbai)	gcp
Enterprise account	jp30	Google Cloud	cf-jp30	Japan (Osaka)	gcp
Enterprise account	sa30	Google Cloud	cf-sa30	KSA (Dammam)	gcp
Enterprise account	il30	Google Cloud	cf-il30	Israel (Tel Aviv)	gcp
Enterprise account	br30	Google Cloud	cf-br30	Brazil (São Paulo)	gcp
Enterprise account	ap30	Google Cloud	cf-ap30	Australia (Sydney)	gcp

## Cluster Regions

When you enable a Kyma environment for a given subaccount, you must select a plan and region where the cluster is going to be created. Note that there is a number of regions available within each plan. They are all listed in the table:

Cluster Regions

Hyperscaler	Plan ID	Region	Region Name
Microsoft Azure	azure	centralus	US Central (IA)
		eastus	US East (VA)
		westus2	US West (WA)
		northeurope	North EU (Ireland)
		uksouth	UK South (London)
		japaneast	Japan (Tokyo)
		southeastasia	Singapore
		westeurope	Europe (Netherlands)
	trial	australiaeast	Australia (Sydney)
		switzerlandnorth	Switzerland (Zurich)
Amazon Web Services	aws	brazilsouth	Brazil (São Paulo)
		southeastasia	Singapore
		eu-central-1	Europe (Frankfurt)
		eu-west-2	Europe (London)
		ca-central-1	Canada (Montreal)
		sa-east-1	Brazil (São Paulo)
		us-east-1	US East (VA)
		us-west-1	US West (N. California)
		ap-northeast-1	Japan (Tokyo)
		ap-northeast-2	South Korea (Seoul)
Google Cloud	gcp	ap-south-1	India (Mumbai)
		ap-southeast-1	Singapore
		ap-southeast-2	Australia (Sydney)
	trial	eu-central-1	Europe (Frankfurt)
		us-east-1	US East (VA)
	gcp	europe-west3	Europe (Frankfurt)
		us-central1	US Central (IA)
		asia-south1	India (Mumbai)

Hyperscaler	Plan ID	Region	Region Name
		asia-northeast2	Japan (Osaka)
		me-central2	KSA (Dammam)
		me-west1	Israel (Tel Aviv)
		australia-southeast1	Australia (Sydney)
		southamerica-east1	Brazil (São Paulo)

## Load Balancers

Depending on the IaaS Provider, the following Load Balancers are provisioned by default:

Default Load Balancers

IaaS Provider	Default Load Balancer
Microsoft Azure	Standard
Amazon Web Services	Classic Network Load Balancer
Google Cloud	External passthrough Network Load Balancer

For more details on the Load Balancers and their features, check out the official documentation of the respective IaaS provider.

## Related Information

[Create a Kyma Instance \[page 2992\]](#)

[Available Plans in the Kyma Environment \[page 2998\]](#)

[Provisioning and Updating Parameters in the Kyma Environment \[page 3002\]](#)

## 2.2.4 Resilience, High Availability, and Disaster Recovery

SAP has a number of processes in place to support resilience in SAP BTP, and provides different offerings so that you can support the high availability of your applications.

## How SAP Provides Resilience

SAP applies resilience principles when developing, updating, and deploying our SAP BTP applications and services.

SAP BTP provides resilience through the following:

Processes and Offerings	Description	Regional Availability
<b>Availability Zones</b>	To achieve better fault-tolerance in the Cloud Foundry environment, we deploy our services across multiple AZs, which improves the availability of a service if there are issues with the infrastructure of one AZ. For more information, see <a href="#">Availability Zones in the Cloud Foundry Environment [page 52]</a> .	All regions that support the Cloud Foundry runtime. See <a href="#">Regions and API Endpoints</a> Available for the Cloud Foundry Environment [page 20].
<b>Backups in Kyma runtime</b>	Kyma runtime relies on managed Kubernetes clusters for periodic backups of Kubernetes objects. For more information, see <a href="#">Kyma Environment Backup [page 3015]</a> .	All regions that support the Kyma runtime. See <a href="#">Regions for the Kyma Environment [page 41]</a> .
<b>Backup and Recovery for SAP HANA Cloud</b>	If you use SAP HANA Cloud, your SAP HANA Cloud instances are continually backed up to safeguard your database and ensure that it can be recovered speedily. For more information, see <a href="#">Backup and Recovery</a> .	All regions where SAP HANA Cloud is available. See <a href="#">Availability of SAP HANA Cloud</a> .
<b>Disaster Recovery</b>	The SAP BTP Disaster Recovery (DR) Plan is part of the overall SAP BTP Business Continuity Plan, which includes crisis management and process continuity activities that are triggered by a declared disaster. For more information, see <a href="#">Disaster Recovery as Part of the Business Continuity Plan [page 47]</a> .	All regions.

## Best Practices for Resilient Applications

In addition to the services offered by SAP BTP, you can follow our best practices for developing and deploying applications, which allow you to make your application running on SAP BTP stable and highly available.

- **Develop Resilient Applications**

When developing your applications, apply the principles and patterns of resilient software design that fit your use case. For more information, see [Developing Resilient Apps on SAP BTP](#). For information specifically about developing applications in the Kyma runtime, see [Develop Resilient Applications in the Kyma Runtime](#). For situations where the load is highly available and where applications need to react by scaling, consider using Application Autoscaler. For more information, see [What Is Application Autoscaler?](#)

- **Working with Availability Zones**

To benefit from the high availability mechanisms in Cloud Foundry, set up your applications with multiple instances. For more information, see [Developing Resilient Applications](#).

### 2.2.4.1 Disaster Recovery as Part of the Business Continuity Plan

The cloud platform disaster recovery (DR) plan is part of the overall cloud platform business continuity plan, which includes crisis management and process continuity activities that are triggered by a declared disaster.

#### Standard Disaster Recovery

SAP can restore productive tenants from backups as soon as practicable in case of a disaster resulting in the loss of the primary production data center.

As the magnitude of a disaster is unpredictable, a region might not be restored in a reasonable time. In addition, a new infrastructure might need to be set up at a different location, which might require the purchase and setup of new hardware. Therefore, we can't guarantee any fixed recovery timelines.

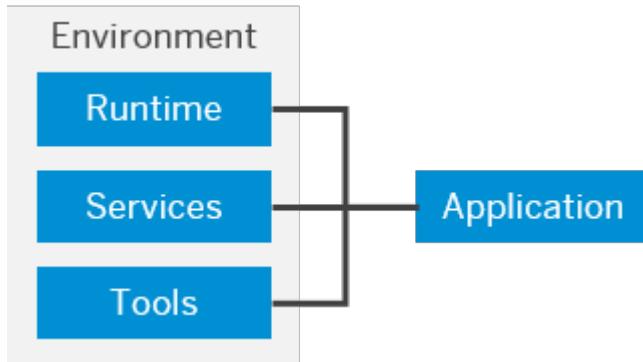
## 2.3 Environments

Environments constitute the actual platform-as-a-service offering of SAP BTP that allows for the development and administration of business applications. Environments are anchored in SAP BTP on subaccount level.

Each environment comes equipped with specific tools, technologies, and runtimes that you need to build applications. So a multi-environment subaccount (for Kyma, ABAP, and Cloud Foundry environments) is your single address to host a variety of applications and offer diverse development options. One advantage of using different environments in one subaccount is that you only need to manage users, authorizations, and entitlements once per subaccount, and thus grant more flexibility to your developers.

## ⓘ Note

The multi-environment subaccount functionality is not applicable for the Neo environment.



## Environment Instances

To actually use an environment in a subaccount, you must **enable** it, which creates an instance of that environment. There are several ways to create **environment instances**:

- In the SAP BTP cockpit, on the subaccount overview, choose [Enable](#).
- In the SAP BTP cockpit, under [Service Marketplace](#). Here, you get more information, such as the available plans and links to further information.
- Using the btp CLI command `btp create accounts/environment-instance`

### [Cloud Foundry Environment \[page 49\]](#)

The Cloud Foundry environment allows you to create polyglot cloud applications in Cloud Foundry. It contains the SAP BTP, Cloud Foundry runtime service, which is based on the open-source application platform managed by the Cloud Foundry Foundation.

### [ABAP Environment \[page 65\]](#)

Within the Cloud Foundry environment, you can create a new space for ABAP development. This is what we refer to as the ABAP environment. It allows you to create extensions for ABAP-based products, such as SAP S/4HANA Cloud, and develop new cloud applications. You can transform existing ABAP-based custom code or extensions to the cloud.

### [Kyma Environment \[page 70\]](#)

SAP BTP, Kyma runtime provides a fully managed cloud-native Kubernetes application runtime based on the open-source project "Kyma". Based on modular building blocks, Kyma runtime includes all the necessary capabilities to simplify the development and to run enterprise-grade cloud-native applications.

### [Neo Environment \[page 81\]](#)

The Neo environment lets you develop HTML5, Java, and SAP HANA extended application services (SAP HANA XS) applications. You can also use the UI Development Toolkit for HTML5 (SAPUI5) to develop rich user interfaces for modern web-based business applications.

## Related Information

[Account Administration \[page 2139\]](#)

### 2.3.1 Cloud Foundry Environment

The Cloud Foundry environment allows you to create polyglot cloud applications in Cloud Foundry. It contains the SAP BTP, Cloud Foundry runtime service, which is based on the open-source application platform managed by the Cloud Foundry Foundation.

The Cloud Foundry environment enables you to develop new business applications and business services, supporting multiple runtimes, programming languages, libraries, and services. You can leverage a multitude of buildpacks, including community innovations and self-developed buildpacks. It also integrates with SAP HANA extended application services, advanced model.

For more information about Cloud Foundry, see the official Cloud Foundry documentation at <https://docs.cloudfoundry.org/>.

## Related Information

[Getting Started in the Cloud Foundry Environment \[page 147\]](#)

[Development in the Cloud Foundry Environment \[page 233\]](#)

[Administration and Operations in the Cloud Foundry Environment \[page 2422\]](#)

## 2.3.1.1 Supported and Unsupported Cloud Foundry Features

Find out which Cloud Foundry features the Cloud Foundry environment on SAP BTP supports and doesn't support.

Supported Features	Unsupported Features
<ul style="list-style-type: none"><li>Diego runtime. See <a href="https://docs.cloudfoundry.org/concepts/diego/diego-architecture.html">https://docs.cloudfoundry.org/concepts/diego/diego-architecture.html</a>.</li><li>SSH. See <a href="https://docs.cloudfoundry.org/devguide/deploy-apps/app-ssh-overview.html">https://docs.cloudfoundry.org/devguide/deploy-apps/app-ssh-overview.html</a>.</li><li>Custom Domains. See <a href="https://docs.cloudfoundry.org/devguide/deploy-apps/routes-domains.html#domains">https://docs.cloudfoundry.org/devguide/deploy-apps/routes-domains.html#domains</a>.</li><li>Docker. See <a href="https://docs.cloudfoundry.org/admin-guide/docker.html">https://docs.cloudfoundry.org/admin-guide/docker.html</a>.</li><li>Running Tasks. See <a href="https://docs.cloudfoundry.org/dev-guide/using-tasks.html">https://docs.cloudfoundry.org/dev-guide/using-tasks.html</a>.</li><li>Request Tracing<ul style="list-style-type: none"><li>Zipkin Tracing. See <a href="https://docs.cloudfoundry.org/adminguide/zipkin_tracing.html">https://docs.cloudfoundry.org/adminguide/zipkin_tracing.html</a>.</li></ul></li><li>Websockets. See <a href="https://docs.cloudfoundry.org/adminguide/supporting-websockets.html">https://docs.cloudfoundry.org/adminguide/supporting-websockets.html</a>.</li><li>Space-Scoped Service Brokers. See <a href="https://docs.cloudfoundry.org/services/managing-service-brokers.html">https://docs.cloudfoundry.org/services/managing-service-brokers.html</a>.</li><li>Route Services (only user-provided and fully-brokered services). See <a href="https://docs.cloudfoundry.org/services/route-services.html">https://docs.cloudfoundry.org/services/route-services.html</a>.</li><li>Sharing Service Instances (not all services support instance sharing). See <a href="https://docs.cloudfoundry.org/devguide/services/sharing-instances.html">https://docs.cloudfoundry.org/devguide/services/sharing-instances.html</a>.</li><li>HTTP/2. See <a href="https://docs.cloudfoundry.org/admin-guide/supporting-http2.html#application">https://docs.cloudfoundry.org/admin-guide/supporting-http2.html#application</a>.</li><li>Streaming Logs to Log Management Services. See <a href="https://docs.cloudfoundry.org/dev-guide/services/log-management.html">https://docs.cloudfoundry.org/dev-guide/services/log-management.html</a>.</li></ul>	<ul style="list-style-type: none"><li>Container-to-Container Networking. See <a href="https://docs.cloudfoundry.org/concepts/understand-cf-networking.html">https://docs.cloudfoundry.org/concepts/understand-cf-networking.html</a>.</li><li>Isolation Segments. See <a href="https://docs.cloudfoundry.org/adminguide/isolation-segments.html">https://docs.cloudfoundry.org/adminguide/isolation-segments.html</a>.</li><li>TCP Routing. See <a href="https://docs.cloudfoundry.org/adminguide/enabling-tcp-routing.html">https://docs.cloudfoundry.org/adminguide/enabling-tcp-routing.html</a>.</li><li>Secure Service Credential Delivery (with Credhub). See <a href="https://docs.cloudfoundry.org/credhub/index.html">https://docs.cloudfoundry.org/credhub/index.html</a> or <a href="https://github.com/cloudfoundry/credhub/blob/main/docs/secure-service-credentials.md">https://github.com/cloudfoundry/credhub/blob/main/docs/secure-service-credentials.md</a>.</li></ul>

## 2.3.1.2 SAP BTP-Specific Configurations

The following technical configurations are specific to SAP BTP and differ from the default configuration:

- SAP BTP supports the Cloud Foundry command line interface (CF CLI) version 8 or newer. Older versions of the CF CLI are not supported.

- By default, a newly pushed (or started) Cloud Foundry application needs to respond to a health check within the first 60 seconds, otherwise the application is considered to have failed. For more information, see [https://docs.cloudfoundry.org/devguide/deploy-apps/healthchecks.html#health\\_check\\_timeout](https://docs.cloudfoundry.org/devguide/deploy-apps/healthchecks.html#health_check_timeout). On SAP BTP, however, you can override this timeout to up to 10 minutes. For instructions, see <https://docs.cloudfoundry.org/devguide/deploy-apps/large-app-deploy.html>.
- On SAP BTP, application SSH access is disabled by default. For more information on SSH, see <https://docs.cloudfoundry.org/devguide/deploy-apps/app-ssh-overview.html>.
- SAP BTP supports the Cloud Foundry API version 3. The Cloud Foundry API v2 has been deprecated and is no longer supported. For more information, see <https://v3apidocs.cloudfoundry.org/>.
- On SAP BTP, the Cloud Foundry API is protected by a rate limit against misuse. The limit is in the range of a few 10k requests per hour per user on average. The rate limit for the deprecated Cloud Foundry API v2 is in the range of a few hundred requests per hour per user.
- In addition to the general rate limit on the Cloud Foundry API, requests for certain API endpoints related to services face a separate limit on concurrent requests. The Cloud Foundry API responds with HTTP status code 429 if a rate limit is reached and provides a Retry-After Header suggesting when the client can attempt a retry. For more information, see <https://docs.cloudfoundry.org/running/rate-limit-cloud-controller-api.html#Rate%20Limit%20Responses:%20Service%20Brokers>.
- In the SAP BTP, Cloud Foundry environment, the total HTTP Request Header and HTTP Response Header size is limited to 64 KB to protect against misuse.
- In the SAP BTP, Cloud Foundry environment, for both HTTP Request Headers and HTTP Response Headers the total amount of Headers is limited to 101.
- In the SAP BTP, Cloud Foundry environment, the limit of concurrent HTTP connections between client and application is 3000 per application container.
- In the SAP BTP, Cloud Foundry environment, the HTTP keep-alive timeout towards the client is set to 60s to protect against misuse. 60s is the maximum time span allowed to wait for a new HTTP request to appear if keep-alive is enabled.
- In the SAP BTP, Cloud Foundry environment, an internal HTTP keep-alive is set to 90s. A higher value must be set on application-side to avoid intermittent disruptions. For more information, see step 4 of [3406978](#).
- In the Cloud Foundry environment, there's a logging rate limit to guard against malicious applications. By default, the limit is 4000 logs per second per application instance, but in exceptional high load scenarios it may be lowered by the platform. If this limit is exceeded, additional logs from the application instance are dropped and a warning message is injected into the application instance's log stream every second. This message also contains the exact current log rate limit.
- Applications requiring sent envelopes to be delivered to external Log Management Services should use the Cloud Foundry syslog drain capability. See <https://docs.cloudfoundry.org/devguide/services/log-management.html>.
- In the SAP BTP, Cloud Foundry environment, the time between signaling a container to shut down gracefully and forcefully stopping it is set to 60 seconds. The default in Cloud Foundry is 10 seconds, see <https://docs.cloudfoundry.org/devguide/deploy-apps/app-lifecycle.html#shutdown>. This time interval will not be taken into account if there are no explicit kernel signal handlers implemented in the application.
- In the SAP BTP, Cloud Foundry environment, applications get a guaranteed CPU share of  $\frac{1}{4}$  core per GB instance memory. As the maximum instance memory per application is 16 GB, this allows for vertical scaling up to 4 CPUs.  
If applications running on the same virtual machine don't use their guaranteed CPU, other applications might get more CPU. This isn't guaranteed and might be subject to change in the future. If you encounter performance problems, scale up your application or increase the application start timeout.

The number of running threads per application instance is limited to 10,420. Reaching this limit can cause performance issues.

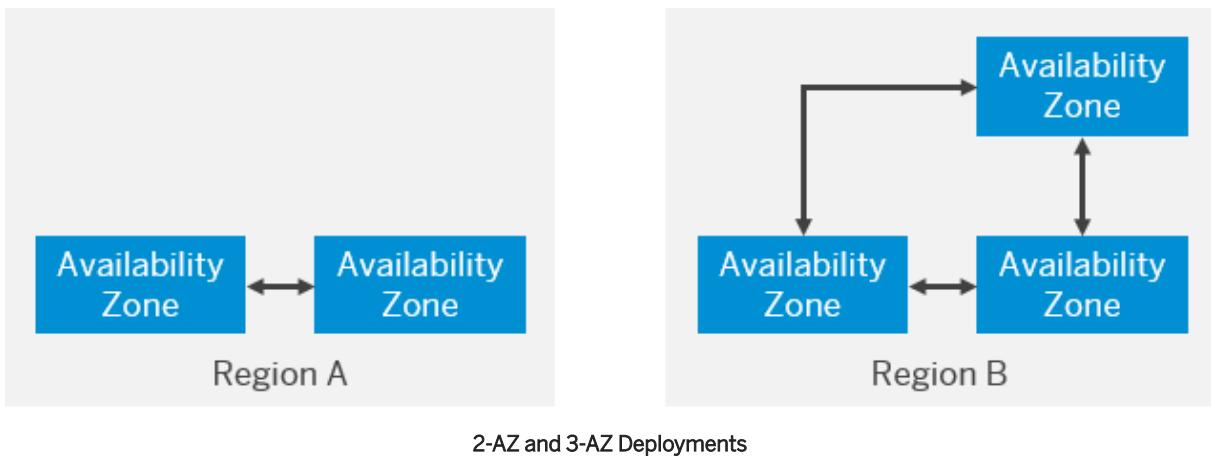
- When pushing or scaling your application, you can define a `disk_quota` that can be up to 10 GB. For more information, see <https://docs.cloudfoundry.org/devguide/deploy-apps/manifest-attributes.html#disk-quota>.
- When deploying applications on SAP BTP, the maximum application package size is 1.5 GB. If your application is larger than that, the deployment fails. For more information, see <https://docs.cloudfoundry.org/devguide/deploy-apps/large-app-deploy.html>.
- In the SAP BTP, Cloud Foundry environment, the hard limit for open file descriptors is 32,768 (32K) per container.
- In global accounts that support the consumption-based commercial model you might see a quota limit for certain services. This is a technical limit only, not a business limit. If you need to increase this limit, report an incident to [SAP support](#) for component BC-NEO-CIS.
- In the SAP BTP, Cloud Foundry environment, the SAP HANA database supports up to 1000 simultaneous connections per database.
- In the SAP BTP, Cloud Foundry environment, each application can be mapped to approximately 1000 routes (128 KB). The total length of the routing information must not exceed this limit.
- Cloud Foundry Audit Events have a retention period of 14 days. For more information on Audit Events, see <https://docs.cloudfoundry.org/running/managing-cf/audit-events.html>.

### 2.3.1.3 Availability Zones in the Cloud Foundry Environment

The Cloud Foundry environment follows the recommendations of our partner IaaS providers by leveraging the availability zones (AZ) concept.

#### About Availability Zones

**Availability zones** (AZ) are single failure domains within a single geographical region and are separate physical locations with independent power, network, and cooling. Multiple AZs exist in one region and are connected with each other through a low-latency network.



To achieve better fault-tolerance, our partners recommend deploying services across multiple AZs, which improves the availability of a service if there are issues with one AZ.

## High Availability at Platform and Application Level

The SAP BTP Cloud Foundry environment follows these recommendations to support high availability at the platform and application level:

- **High availability of the platform components:**
  - The building blocks of Cloud Foundry and the virtual machines on which the Cloud Foundry application instances are scheduled run in a high availability setup. Their instances are distributed across different AZs.
  - The technology that manages the deployment of the Cloud Foundry environment monitors the health of the platform. If there are infrastructure failures, it re-creates the faulty components.
- **High availability on the application level:**
  - We recommend running multiple application instances to increase availability. For more information, see [Run Multiple Instances to Increase Availability](#). On SAP BTP, there are three ways to increase application instances:
    - Scaling your application using the application manifest. The `manifest.yml` allows you to make and save configurations for your application. To scale, you can configure the instance count in the manifest and push the application again with the new configuration. See [App Manifest Attribute Reference](#). To avoid downtimes when updating your application configuration, you can also consider using rolling application deployments. See [Rolling App Deployments](#).
    - Scaling your application using the `cf scale` command in the Cloud Foundry command line interface (CF CLI). See [Scaling an App Using cf scale](#).
    - Scaling your application using the SAP BTP cockpit. See [Add or Remove Application Instances](#).
  - The Cloud Foundry container scheduler takes care of distributing the different instances of one application on virtual machines in different AZs. For more information, see [How Diego Balances App Processes](#).
  - Cloud Foundry is constantly monitoring the health state of application instances and restarts instances that are considered unhealthy. See [Using App Health Checks](#).

- When the number of desired instances doesn't match the number of actually running instances, Cloud Foundry reschedules the missing instances, for example, when the virtual machines that an application instance was initially scheduled on become unresponsive.

For more information on high availability configuration, see [High Availability in Cloud Foundry](#).

For more information on application stability and resilience, see [Developing Resilient Applications](#).

## 2.3.1.4 Additional Information About Cloud Foundry

Links to additional information about Cloud Foundry that is useful to know but not necessarily directly connected to the SAP BTP, Cloud Foundry environment.

Content	Location
BOSH	<a href="http://bosh.cloudfoundry.org">http://bosh.cloudfoundry.org</a>
BOSH documentation	<a href="http://bosh.io/docs">http://bosh.io/docs</a>
Buildpacks	<a href="http://docs.cloudfoundry.org/buildpacks">http://docs.cloudfoundry.org/buildpacks</a>
Components of Cloud Foundry	<a href="http://docs.cloudfoundry.org/concepts/architecture/">http://docs.cloudfoundry.org/concepts/architecture/</a>
Cloud Foundry Concepts	<a href="http://docs.cloudfoundry.org/concepts/">http://docs.cloudfoundry.org/concepts/</a>
Deployment of Cloud Foundry	<a href="http://docs.cloudfoundry.org/deploying">http://docs.cloudfoundry.org/deploying</a>
Developer Guide for Cloud Foundry	<a href="http://docs.cloudfoundry.org/devguide">http://docs.cloudfoundry.org/devguide</a>
Diego Application Process Balancing	<a href="https://docs.cloudfoundry.org/concepts/diego/diego-auction.html">https://docs.cloudfoundry.org/concepts/diego/diego-auction.html</a>
Glossary for Cloud Foundry	<a href="http://docs.cloudfoundry.org/concepts/glossary.html">http://docs.cloudfoundry.org/concepts/glossary.html</a>
Overview of Cloud Foundry	<a href="http://docs.cloudfoundry.org/concepts/overview.html">http://docs.cloudfoundry.org/concepts/overview.html</a>
Sample applications for Cloud Foundry	<a href="https://github.com/cloudfoundry-samples">https://github.com/cloudfoundry-samples</a>
Security settings for Cloud Foundry	<a href="http://docs.cloudfoundry.org/concepts/security.html">http://docs.cloudfoundry.org/concepts/security.html</a>
Cloud Foundry Services	<a href="http://docs.cloudfoundry.org/services">http://docs.cloudfoundry.org/services</a>
	<a href="http://docs.cloudfoundry.org/devguide/services/user-provided.html">http://docs.cloudfoundry.org/devguide/services/user-provided.html</a>
Considerations for designing and running an application in the cloud	<a href="http://docs.cloudfoundry.org/devguide/deploy-apps/prepare-to-deploy.html">http://docs.cloudfoundry.org/devguide/deploy-apps/prepare-to-deploy.html</a>
Installing the Cloud Foundry command line interface	<a href="http://docs.cloudfoundry.org/devguide/installcf/install-go-cli.html">http://docs.cloudfoundry.org/devguide/installcf/install-go-cli.html</a>

Content	Location
Blog about Cloud Foundry	<a href="http://blog.cloudfoundry.org/">http://blog.cloudfoundry.org/</a>

## 2.3.1.5 Service Plans and Metering for Cloud Foundry Runtime

This page explains the relationship between the service plans in the SAP Discovery Center and those in the SAP BTP cockpit, and provides information to help you understand how the service is billed.

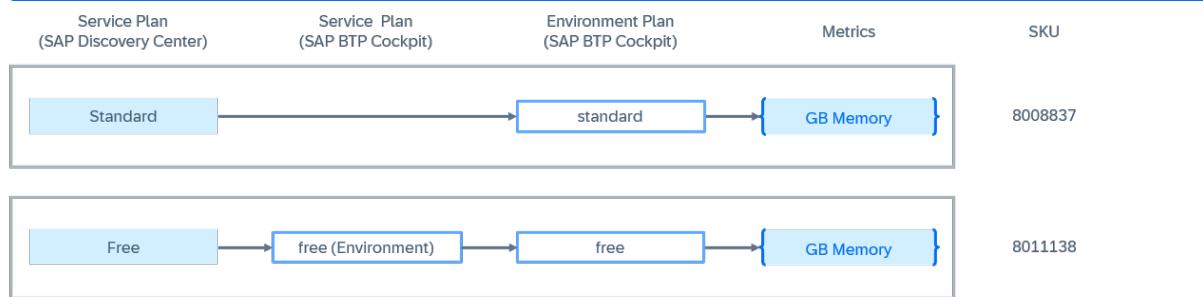
### Service

#### Overview

The diagram below shows how the service plans listed in the [SAP Discovery Center](#) correspond to the plans you choose in the [SAP BTP cockpit](#), depending on the commercial model of your enterprise global account. For more information about the commercial models offered by SAP, see [Commercial Models \[page 86\]](#).

This image is interactive. Hover over rectangles in the diagram to see the description. Choose the highlighted areas for more information.

#### Consumption-Based Commercial Model



#### Subscription-Based Commercial Model

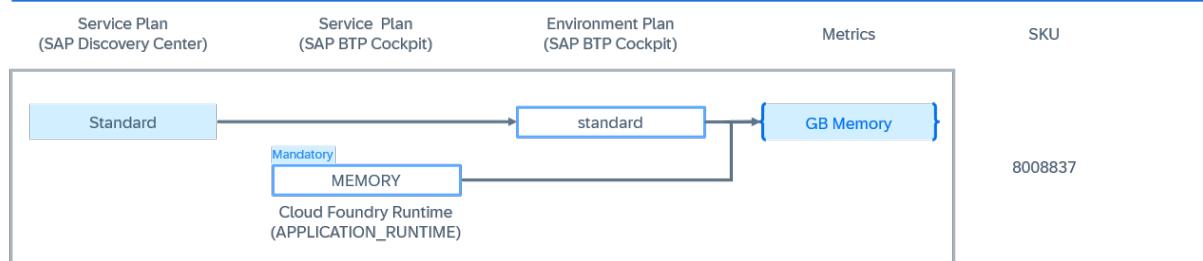


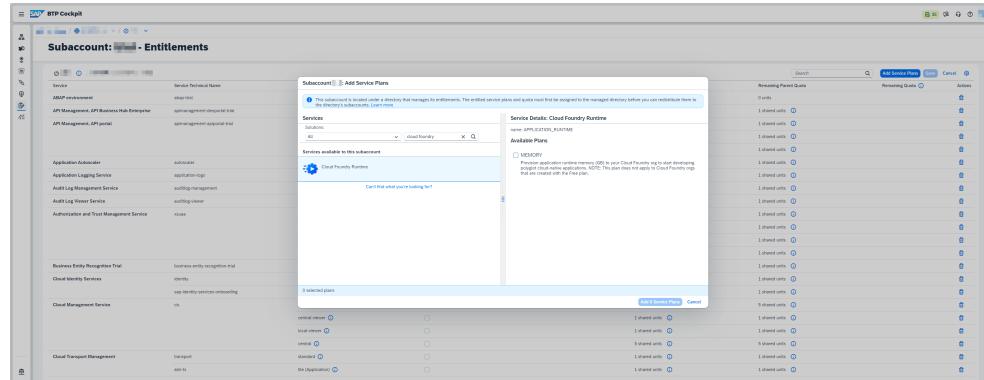
Diagram captions: Service plans and environment plans in SAP BTP cockpit [Show/Hide]

## Service Plan (SAP BTP Cockpit)

The plan that the global account administrator can assign to a subaccount or a directory on the [Entitlements](#) page by choosing ► [Edit](#) ► [Add Service Plans](#) ▶.

→ Tip

Open image in new tab for the full-screen version.



Screenshot: Assigning a service plan to a subaccount in SAP BTP cockpit [Show/Hide]

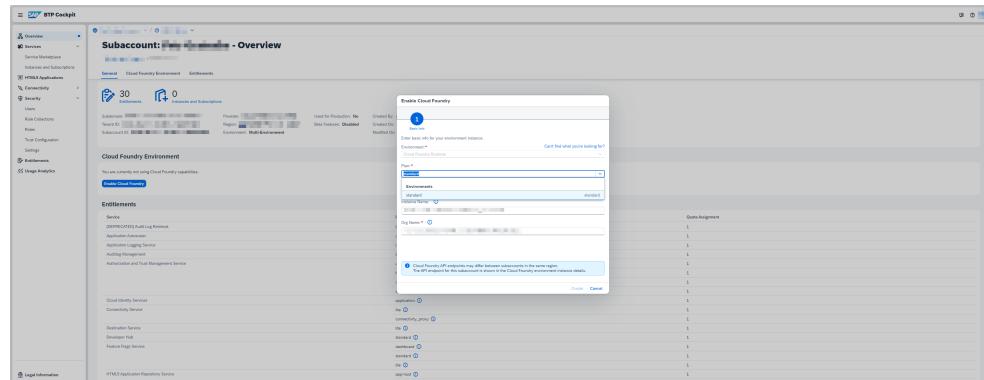
For more information about the procedure, see [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#) or [Configure Entitlements and Quotas for Directories \[page 2188\]](#).

## Environment Plan (SAP BTP Cockpit)

The plan that the subaccount administrator can select on the [Overview](#) page of a subaccount when choosing [Enable Cloud Foundry](#).

→ Tip

Open image in new tab for the full-screen version.



Screenshot: Selecting an environment plan for the subaccount in SAP BTP cockpit [Show/Hide]

For more information about the procedure, see [Create Orgs \[page 2435\]](#).

## ⓘ Note

An environment plan is a service plan associated with an environment. For more information about services and service plans, see [Entitlements and Quotas \[page 100\]](#).

## Service Plans

The tables below provide details about the plans for SAP BTP, Cloud Foundry runtime. They can give you more context for understanding the diagram in the **Overview** section.

Table 1: Plans for Consumption-Based Commercial Model [Show/Hide]

Service Plan (SAP Discovery Center)	Service (SAP BTP Cockpit: <i>Entitlements</i> )	Service Plan (SAP BTP Cockpit: <i>Entitlements</i> )	Environment Plan (SAP BTP Cockpit: <i>Enable Cloud Foundry</i> )	Explanation
Standard	-	-	standard	<p>This is a <b>paid</b> plan for productive use. In the consumption-based commercial model, you are charged based on how much runtime memory has been consumed by your applications running in the Cloud Foundry environment. For details, see <a href="#">Service Specifications [page 63]</a>.</p> <p><b>⚠ Caution</b></p> <p>With this plan, you get a technical quota of 200 GB of runtime memory per Cloud Foundry org. This doesn't mean that you can use the runtime memory for free. The technical quota represents a <b>limit</b> on how much runtime memory all the spaces in the org can use at any given time. To increase this limit, create a case on the <a href="#">SAP Support Portal</a> using the component BC-NEO-CIS.</p> <p>The plan is available for all subaccounts by default and can be enabled by a subaccount administrator.</p>

Service Plan (SAP Discovery Center)	Service (SAP BTP Cockpit: <i>Entitlements</i> )	Service Plan (SAP BTP Cockpit: <i>Entitlements</i> )	Environment Plan (SAP BTP Cockpit: <i>Enable Cloud Foundry</i> )	Explanation
Free	Cloud Foundry Environment (cloudfoundry)	free (Environment)	free	<p>This is a <b>free tier</b> plan that allows you to try out and evaluate the service. For more information, see <a href="#">Using Free Service Plans [page 91]</a>.</p> <p>With this plan, you get a free quota of runtime memory for your Cloud Foundry org. The amount of such free quotas per global enterprise account is limited.</p> <p>The plan must be assigned to the subaccount by a global account administrator (or a directory administrator), before it can be enabled by a subaccount administrator.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>Only community support is available for free tier service plans and these are not subject to SLAs. Use of free tier service plans is subject to additional terms and conditions as provided in the <a href="#">Business Technology Platform Supplemental Terms and Conditions</a>.</p> </div>

Service Plan (SAP Discovery Center)	Service (SAP BTP Cockpit: <i>Entitlements</i> )	Service Plan (SAP BTP Cockpit: <i>Entitlements</i> )	Environment Plan (SAP BTP Cockpit: <i>Enable Cloud Foundry</i> )	Explanation
See <a href="#">SAP Build Code</a>	Cloud Foundry Environment (cloudfoundry)	build-code	build-code	<p>This plan is only for using the SAP BTP, Cloud Foundry runtime as part of SAP Build Code. For more information, see <a href="#">What Is SAP Build Code</a>.</p> <div style="background-color: #e0e0e0; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>The plan is not depicted on the diagram in the <b>Overview</b> section.</p> </div>

Table 2: Plans for Subscription-Based Commercial Model [Show/Hide]

Service Plan (SAP Discovery Center)	Service (SAP BTP Cockpit: <i>Entitlements</i> )	Service Plan (SAP BTP Cockpit: <i>Entitlements</i> )	Environment Plan (SAP BTP Cockpit: <i>Enable Cloud Foundry</i> )	Explanation
Standard	Cloud Foundry Runtime (APPLICATION_RUNTIME)	MEMORY	standard	<p>This is a <b>paid</b> plan for productive use. In the subscription-based commercial model, you pay for runtime memory quota in advance.</p> <p>The runtime memory quota purchased with your subscription must be entitled to subaccounts through the service plan MEMORY. This service plan is associated with the service Cloud Foundry Runtime (APPLICATION_RUNTIME), which is specific to the subscription-based commercial model.</p> <p>The plan <b>standard</b> is available for all subaccounts by default and can be enabled by a subaccount administrator. The runtime memory quota can be entitled to the subaccount only by a global account administrator (or a directory administrator). These steps are independent, but both of them are required to run applications in the Cloud Foundry environment.</p>

Service Plan (SAP Discovery Center)	Service (SAP BTP Cockpit: <i>Entitlements</i> )	Service Plan (SAP BTP Cockpit: <i>Entitlements</i> )	Environment Plan (SAP BTP Cockpit: <i>Enable Cloud Foundry</i> )	Explanation
See <a href="#">SAP Build Code</a>	Cloud Foundry Environment ( <code>cloudfoundry</code> )	build-code	build-code	<p>This plan is only for using the SAP BTP, Cloud Foundry runtime as part of SAP Build Code. For more information, see <a href="#">What Is SAP Build Code</a>.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>The plan is not depicted on the diagram in the <b>Overview</b> section.</p> </div>

## Metrics

The table below provides details about the metrics for SAP BTP, Cloud Foundry runtime. It includes the names of services, with which the metrics are associated on the pages [Usage](#) (accessed from the global account level) and [Usage Analytics](#) (accessed from the subaccount level) in the SAP BTP cockpit.

Service	Metric	Definition	Additional Information
Cloud Foundry Runtime	GB Memory	<p>Temporary memory bank where computers store data that needs to be retrieved and processed quickly.</p> <p>The memory represents the size of the data that can be processed and CPU represents the speed at which the data can be retrieved.</p>	<p>For billing purposes, the metric GB Memory is calculated as the total hourly usage of Cloud Foundry runtime memory across all spaces in the global account over a calendar month, divided by 730 hours and rounded up to the next full GB. For an example of such calculation, see <a href="#">Consumption Monitoring</a>.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>In the Cloud Foundry environment, applications get a guaranteed CPU share of <math>\frac{1}{4}</math> core per GB of runtime memory quota reserved for an application instance. For more information, see <a href="#">SAP BTP-Specific Configurations [page 50]</a>.</p> </div>

Service	Metric	Definition	Additional Information
SAP Build Code	CF Runtime	The metric CF Runtime is intended only for monitoring the usage of SAP BTP, Cloud Foundry runtime when it's used as part of SAP Build Code. For more information, see <a href="#">What Is SAP Build Code</a> .	<p><b> ⓘ Note</b></p> <p>The metric is not depicted on the diagram in <a href="#">Service [page 55]</a>.</p>

## Service Specifics

In the context of SAP BTP, Cloud Foundry runtime, the terms **consumption** (or **consume**) and **usage** (or **use**) refer to the runtime memory quota **reserved** by the platform for each application instance. This quota serves as the basis for calculating the billable consumption, as explained in [Consumption Monitoring](#).

### ⚠ Caution

From a billing standpoint, it doesn't matter how much of the reserved runtime memory quota is utilized when the application is running. Billable consumption is calculated based on the full amount of runtime memory reserved, regardless of how much runtime memory actually gets utilized.

### ⓘ Note

If an application is stopped, it doesn't reserve any runtime memory and, therefore, doesn't contribute to runtime memory consumption.

There are two application settings that define how much runtime memory an application uses at any given time:

- Runtime memory quota reserved for each application instance (default: 1024 MB)
- Number of running application instances (default: 1)

You can specify custom values for these settings when deploying an application in the Cloud Foundry environment. You can also change them for an application that has already been deployed without having to redeploy it.

### How to set runtime memory quota and number of instances when deploying applications

- **Option 1:** Create a manifest YAML file and specify the following application attributes:
    - `memory` ↗ : runtime memory quota per application instance
    - `instances` ↗ : number of application instances
- For more information about the manifest formatting, see <https://docs.cloudfoundry.org/devguide/deploy-apps/manifest-attributes.html> ↗ .

You can then deploy your application with the manifest in one of two ways:

- Using the SAP BTP cockpit: [Deploy an Application \[page 2491\]](#)
- Using the Cloud Foundry CLI: <https://docs.cloudfoundry.org/devguide/deploy-apps/manifest.html>
- **Option 2:** In the SAP BTP cockpit, choose *Deploy Application*, deselect *Use Manifest* and specify the following:
  - *Instance Memory (MB)*: runtime memory quota per application instance
  - *Number of Instances*: number of application instances
 For more information about the procedure, see [Deploy an Application \[page 2491\]](#).
- **Option 3:** In the Cloud Foundry CLI, use the command `cf push` with the following flags:
  - `-m`: runtime memory quota per application instance
  - `-i`: number of application instances
 For more information about the procedure, see <https://docs.cloudfoundry.org/devguide/deploy-apps/deploy-app.html#custom-push>.

## How to change runtime memory quota and number of instances for already deployed applications

- **Option 1:** In the SAP BTP cockpit, on the *Overview* page of the application you can:
  - Change the runtime memory quota per application instance under [Change Instance Details](#)
  - Change the number of application instances as described in [Add or Remove Application Instances \[page 2494\]](#)
- **Option 2:** In the Cloud Foundry CLI, you can use the command `cf scale` with the following flags:
  - `-m`: change the runtime memory quota per application instance
  - `-i`: change the number of application instances
 For more information about the procedure, see <https://docs.cloudfoundry.org/devguide/deploy-apps/cf-scale.html>.

You can use the **Application Autoscaler** to automatically increase or decrease the number of application instances based on the policies you have defined. For more information, see [What Is Application Autoscaler](#).

### → Tip

For tips on how to optimize the consumption of runtime memory, see the blog post [Optimise your SAP BTP, Cloud Foundry runtime costs](#). Note that while the general principles outlined in the blog post still apply, some of the UI texts and parameter names may have changed.

## Supplemental Terms and Conditions

For more information, see the section **SAP BTP, Cloud Foundry Runtime** in the [SAP Business Technology Platform Service Description Guide](#).

## Glossary

### [Commercial Information Glossary](#)

## 2.3.2 ABAP Environment

Within the Cloud Foundry environment, you can create a new space for ABAP development. This is what we refer to as the ABAP environment. It allows you to create extensions for ABAP-based products, such as SAP S/4HANA Cloud, and develop new cloud applications. You can transform existing ABAP-based custom code or extensions to the cloud.

The ABAP environment is based on the latest ABAP platform cloud release that is also used for SAP S/4HANA Cloud. It leverages the innovations provided by SAP HANA. The software stack contains standard technology components that are familiar from the standalone Application Server ABAP. The ABAP environment supports the ABAP RESTful Application Programming Model including SAP Fiori and Core Data Services (CDS). SAP services and APIs are offered according to the new approach of released objects. The ABAP environment can be integrated with other SAP BTP services, such as SAP Destination service, SAP Build Work Zone, standard edition, SAP Workflow Management, and SAP Interactive Forms by Adobe.

Each ABAP system in the ABAP environment utilizes a dedicated SAP HANA database, which is provided by the SAP HANA Cloud service and managed by the ABAP environment. The database is linked 1:1 to the ABAP system.

For information about regional availability, see [Regions and API Endpoints for the ABAP Environment \[page 37\]](#).

### Related Information

[Getting Started in the ABAP Environment \[page 165\]](#)

[Development in the ABAP Environment \[page 705\]](#)

[Administration and Operations in the ABAP Environment \[page 2536\]](#)

Using Free Service Plans

[Discovery Center!\[\]\(86f17b72e5a99c99f2f6adb6b83c466b\_img.jpg\)](#)

[SAP Community!\[\]\(fa4d985e7ef3d64a1d6025ab1548d218\_img.jpg\)](#)

[SAP Road Map Explorer!\[\]\(5f99a9243d548108cb02a04439111373\_img.jpg\)](#)

[Learning Journey](#)

[Tutorials!\[\]\(9c804f93ace0e6359980d1fd75689fea\_img.jpg\)](#)

### 2.3.2.1 ABAP Development Tools for Eclipse

ABAP development tools for Eclipse is SAP's state-of-the-art integrated development environment (IDE) for ABAP development on the open Eclipse platform.

### Features

ABAP development tools for Eclipse enables you to perform ABAP-based development tasks, when you want to build, to extend, and to run ABAP applications based on SAP products, such as SAP S/4HANA and SAP

S/4HANA Cloud, private edition for classic ABAP development as well as SAP BTP, ABAP environment and SAP S/4HANA Cloud for Cloud development.

ABAP development tools for Eclipse supports ABAP developers with the following possibilities:

- Highly flexible, customizable Eclipse UI for ABAP development tools
- High-performance, failover-safe online development in multiple ABAP systems simultaneously
- Display and edit functionality for multiple ABAP objects in parallel
- Advanced and efficient source code editing including refactoring support
- Optimum support of task-oriented and test-driven development
- Robust and reliable quality assurance and supportability tools.
- Built-in extensibility of the IDE using the established Eclipse plug-in technology

In particular, ABAP development tools for Eclipse is a modern development toolset where ABAP developers can use, for example the following features:

- Syntax check
- Code completion
- Syntax highlighting
- Pretty printing
- Navigation
- Search
- Quick fixes
- and many more

## Tools User Guides

The [ABAP Development Tools: User Guide](#) describes the functionality and usage of the possibilities. It focuses on use cases for creating, editing, testing, debugging, and profiling development objects.

The [ABAP CDS Development Tools: User Guide](#) describes the functionality and usage of tools for ABAP Core Data Services (CDS) in the ABAP environment. It focuses on use cases for creating, editing, testing, and analyzing ABAP CDS entities.

## Release Notes

ABAP development tools for Eclipse is released to customers in combination with the SAP BTP ABAP environment shipments. Documentation in the context of ABAP Platform will be shipped in accordance with the relevant SAP product shipments.

The release notes are a general overview of the most significant changes relating to features of ABAP development tools for Eclipse in the context of ABAP development and the ABAP programming models.

For more information, see [Release Notes of ABAP Development Tools for Eclipse](#).

## **FAQs**

If you are an SE80 expert and new to ABAP development tools for Eclipse, the frequently asked questions (FAQs) enable you to skim the features you want to perform in ADT.

For more information, see [FAQs for SE80 Experts Using ADT](#).

## **Installation**

To install ABAP development tools for Eclipse, follow the [Installation Guide](#). Alternatively, you may get ABAP development tools for Eclipse from the [SAP Development Tools](#) page under the terms of the [SAP DEVELOPER LICENSE AGREEMENT](#).

## **Support**

The use of ABAP development tools for Eclipse is subject to the terms and conditions of your license agreement with SAP which is directly related to the SAP shipment channel from which ABAP development tools for Eclipse was initially downloaded and installed.

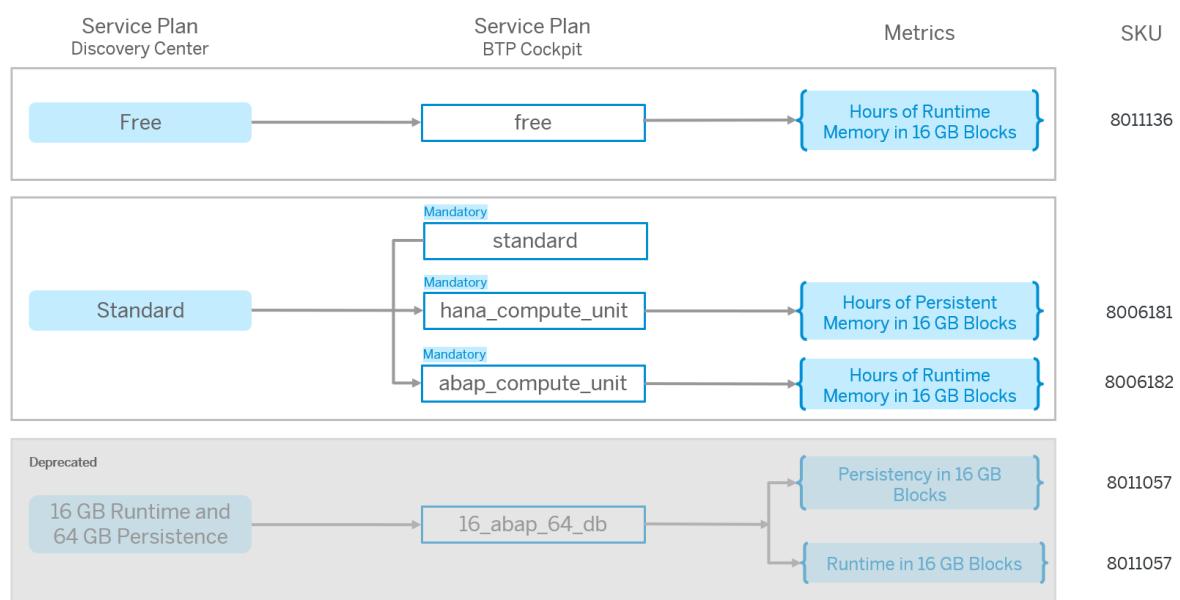
## 2.3.2.2 Commercial Information for SAP BTP, ABAP environment

This page explains the relationship between the service plans of the SAP Discovery Center and the service plans of the SAP BTP cockpit and provides information to help you understand how the service is billed.

### Service

#### Overview

##### Consumption-Based Models



#### Discovery Center: SAP BTP ABAP environment

##### ⓘ Note

For more information about consumption-based models, please check [What is the Consumption-Based Commercial Model?](#)

## SAP BTP Cockpit: Service Plans

Name	Service Plan	Description
free	Free	<p>Allows you to create small proof-of-concept development projects. Also can be used for the piloting of a remote ABAP test cockpit scenario against on-premise systems in the Custom Code Migration app. The instance will be stopped each night automatically and needs to be started again via the Landscape Portal. Only community support is available for free tier service plans and these are not subject to SLAs.</p>
standard	Standard	<p>Allows you to create projects for development, test, and productive ABAP systems. The minimum system size is 2 HANA Compute Units and 1 ABAP Compute Unit.</p> <p>It is mandatory to assign a combination of three entitlement service plans: ABAP Compute Unit, HANA Compute Unit, and either the Free, Standard, or SaaS OEM plan.</p>
abap_compute_unit	Standard	<p>Allows you to configure runtime memory, which refers to volatile memory used during the execution of applications. This memory provides fast read and write access but loses all data when the system is restarted or shut down. The metric measures the amount of runtime memory used in 16 GB blocks on an hourly basis.</p> <p>It is mandatory to assign a combination of three entitlement service plans: ABAP Compute Unit, HANA Compute Unit, and either the Free, Standard, or SaaS OEM plan.</p>
hana_compute_unit	Standard	<p>Allows you to configure persistent memory, which refers to storage that retains data even after the system is restarted or shut down. The metric measures the amount of persistent storage used in 16 GB blocks on an hourly basis.</p> <p>It is mandatory to assign a combination of three entitlement service plans: ABAP Compute Unit, HANA Compute Unit, and either the Free, Standard, or SaaS OEM plan.</p>
saas_oem	SaaS OEM	<p>Allows you to create projects for ABAP systems to run multitenancy SaaS applications. The minimum system size is 2 HANA Compute Units and 1 ABAP Compute Unit.</p> <p>It is mandatory to assign a combination of three entitlement service plans: ABAP Compute Unit, HANA Compute Unit, and either the Free, Standard, or SaaS OEM plan.</p>

Name	Service Plan	Description
16_abap_64_db (Deprecated)	16 GB Runtime and 64 GB Persistence (Deprecated)	<p>This ABAP runtime service plan provides a 16 GB ABAP runtime with a 64 GB database. No additional sizing is possible.</p> <p>This service plan is deprecated. Please change the service plan to "standard".</p>

## Metrics

Metric	Definition
Memory	<p>Temporary memory bank where computers store data that needs to be retrieved and processed quickly.</p> <p>The memory represents the size of the data that can be processed, and CPU represents the speed at which the data can be retrieved.</p>

## Supplemental Terms and Conditions

For more information, see the [SAP Business Technology Platform Service Description Guide](#).

### 2.3.3 Kyma Environment

SAP BTP, Kyma runtime provides a fully managed cloud-native Kubernetes application runtime based on the open-source project "Kyma". Based on modular building blocks, Kyma runtime includes all the necessary capabilities to simplify the development and to run enterprise-grade cloud-native applications.

### Kyma as a Managed Service

Kyma environment permits a native consumption of the Multi-Cloud Foundation Services ([What Is the Multi-Cloud Foundation?](#)) and smooth consumption of SAP and non-SAP applications. It also supports out-of-the-box CAP, SAP Cloud SDK, application router, and HTML5 deployer.

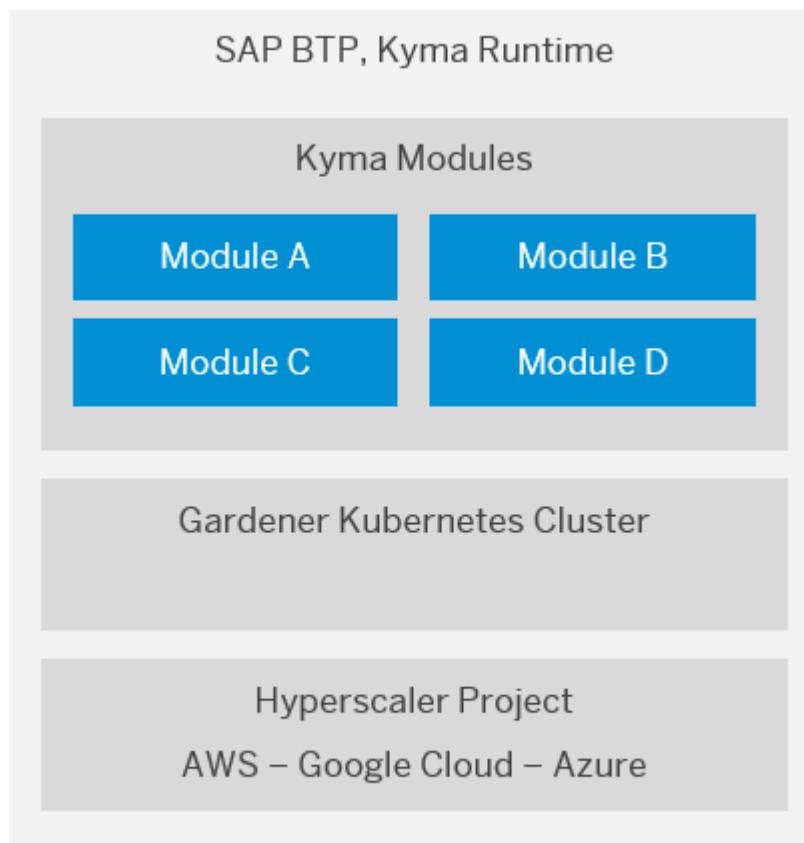
Kyma implements a dedicated application runtime to deploy highly scalable, robust, and secure containerized microservices.

### ⓘ Note

Kyma as a managed service automatically checks all Kyma-managed resources. Any unexpected modifications are discarded, and the resource is reverted to the original state.

Every Kyma environment consists of:

- A Kubernetes cluster based on project "Gardener" on a cloud provider and region (data center) of your choice. To find out the available regions and providers, see [Regions for the Kyma Environment \[page 41\]](#).
- A set of Kyma modules picked by a user in the release channel of their choice installed on the provisioned cluster.



## Integration

Every Kyma environment runs on a single Kubernetes cluster created for a specific subaccount. The configuration of the Kyma environment enables you to connect it to a multitude of SAP systems. This way, you can build various formations that aggregate the SAP systems and environment according to your business use cases.

SAP systems connected to a Kyma environment expose APIs and events. To extend the existing logic of these SAP services, you can build serverless applications called "Functions", and trigger them to react to particular

events or calls to your application's API. You can also use the Kyma environment to deploy microservices or even build full-stack applications.

## Related Information

[Getting Started in the Kyma Environment \[page 223\]](#)

[Development in the Kyma Environment \[page 1864\]](#)

[Administration and Operations in the Kyma Environment \[page 2991\]](#)

[Security in the Kyma Environment \[page 3120\]](#)

[Discovery Center!\[\]\(f063ef9e16f4c52ea7f0906a8473add2\_img.jpg\)](#)

### 2.3.3.1 Kyma's Modular Approach

With Kyma's modular approach, you can install only the modules you need, instead of a predefined set of components.

#### Kyma Module or Component?

Classic Kyma runtime offered a fixed set of mandatory interdependent components whose development rhythm was synchronized and determined by the release schedule. One by one, all Kyma components have been converted to modules that work independently from each other.

The modular approach boosts Kyma's extensibility and scalability and makes it more cost-effective. With fully modularized Kyma, you can choose from many of its modules to facilitate your business needs best. You can add or delete modules on demand and configure them, thus determining the costs generated by the infrastructure that your modules use.

You can decide which modules are needed for your workload, although some features may still require the presence of other modules or related capabilities. For example, the API Gateway module depends on Istio capabilities, but it doesn't require the Istio module from Kyma. Instead, the API Gateway module needs Istio custom resources, such as `Gateway` and `VirtualService`, to be present in the cluster. If you want to use your own Istio installation, the API Gateway module will use it as soon as the required resources are available.

Learn more about the available Kyma modules: [Kyma Modules \[page 74\]](#).

Learn how to add a module under [Add and Delete a Kyma Module \[page 3012\]](#).

#### Kyma Release Channels

Sets of Kyma modules are deployed in two release channels:

- *Regular channel* is the default release channel.
- *Fast channel* provides more frequent releases. It offers early previews of all new features and changes before they are promoted to the regular channel. It also allows you to test and provide feedback on the new features sooner.

According to the Kyma modules' release cycle, we first release a new module's major or minor version in the fast channel. After approximately two weeks, we promote the release to the regular channel.

#### Note

In case of important functionality fixes or critical vulnerabilities identified by our security organization, the timeline doesn't apply, as we provide hotfixes between regular releases.

You can use one or both release channels in your Kyma cluster, but you can define only one release channel per module. For example, you can mix different modules from the regular and fast channels in your development cluster, but you cannot deploy the same module in the regular and fast versions in one cluster.

#### Note

You can upgrade module versions, but you cannot downgrade them. To test the upstream versions, you can switch a module or an entire cluster from the regular channel to the fast one. To return to the regular channel, you must wait until the version you are using in the fast channel is promoted to the regular channel. Once the versions in both the fast and regular channels are the same, you can switch back to regular. Alternatively, you can delete and add your module from the regular channel.

To find out which module version is running in your cluster, go to [Kyma dashboard](#).

## Release Notes

A release of a new module's version is announced with a release note in [What's New for SAP Business Technology Platform](#) for both, the fast and regular channels:

- On the day of the release in the fast channel, a release note is published with the *Preview* label.
- After approximately two weeks, the module version becomes available in the regular channel and the *Preview* label is removed.

## Related Information

[Kyma Functionalities \[page 76\]](#)

## 2.3.3.2 Kyma Modules

With Kyma's modular approach, you can install just the modules you need, instead of a predefined set of components.

You can choose to add any modules as required. To learn how, see [Add and Delete a Kyma Module \[page 3012\]](#). To find out which module version is running in your cluster, go to [Kyma dashboard](#).

### → Tip

A release of a new module's version is announced with a release note in [What's New for SAP Business Technology Platform](#) for both, the fast and regular channels:

- On the day of the release in the fast channel, a release note is published with the *Preview* label.
- When the module version becomes available in the regular channel (after approximately two weeks), the *Preview* label is removed.

## Default Kyma Modules

When you create Kyma runtime in SAP BTP cockpit, it is provisioned with the default modules added. The default modules are not mandatory. If you don't need them, you can delete them in [Kyma dashboard](#). See [Add and Delete a Kyma Module \[page 3012\]](#).

Default Kyma Modules

Module	Technical name	Purpose	Documentation
<i>API Gateway</i>	<code>api-gateway</code>	API Gateway provides functionalities that allow you to expose and secure APIs.	<a href="#">kyma-project.io: API Gateway module</a>
<i>Istio</i>	<code>istio</code>	Istio is a service mesh with Kyma-specific configuration.	<a href="#">Istio Module [page 1868]</a>
<i>SAP BTP Operator</i>	<code>btp-operator</code>	Within the SAP BTP Operator module, BTP Manager installs the SAP BTP service operator that allows you to consume SAP BTP services from your Kubernetes cluster using Kubernetes-native tools.	<a href="#">SAP BTP Operator Module [page 1894]</a> <a href="#">What's New for SAP BTP Operator</a> <a href="#">GitHub repository: BTP Manager</a>

## Optional Kyma Modules

When you create Kyma runtime in SAP BTP cockpit, the following modules are not added by default, but you can choose to add and delete them anytime in [Kyma dashboard](#). See [Add and Delete a Kyma Module \[page 3012\]](#).

## Optional Kyma Modules

Module	Technical name	Purpose	Documentation
<i>Application Connector</i>	<code>application-connector</code>	Application Connector allows you to connect with external solutions. No matter if you want to integrate an on-premise or a cloud system, the integration process doesn't change, which allows you to avoid any configuration or network-related problems.	<a href="#">kyma-project.io: What is Application Connectivity in Kyma?</a> ↗ <a href="#">GitHub repository: Application Connector Manager</a> ↗
<i>Eventing</i>	<code>eventing</code>	<p>Eventing provides functionality to publish and subscribe to CloudEvents.</p> <p>At the moment, the SAP Event Mesh default plan and NATS are supported. If you choose NATS, add the NATS module.</p>	<a href="#">Configure SAP Event Mesh for Kyma Eventing [page 1986]</a> <a href="#">kyma-project.io: Eventing module</a> ↗ <a href="#">GitHub repository: Eventing</a> ↗
<i>Keda</i>	<code>keda</code>	The Keda module comes with Keda Manager, an extension to Kyma that allows you to install <a href="#">KEDA</a> ↗ (Kubernetes Event Driven Autoscaler).	<a href="#">Keda Module [page 1918]</a>
<i>NATS</i>	<code>nats</code>	NATS deploys a NATS cluster within the Kyma cluster. You can use it as a backend for Kyma Eventing.	<a href="#">kyma-project.io: NATS module</a> ↗ <a href="#">GitHub repository: NATS</a> ↗
<i>Serverless</i>	<code>serverless</code>	With the Serverless module, you can define simple code snippets (Functions) with minimal implementation effort.	<a href="#">Deploy Workloads in the Kyma Environment to Extend SAP Systems [page 1977]</a> <a href="#">kyma-project.io: What is Serverless in Kyma?</a> ↗ <a href="#">kyma-project.io: Serverless Configuration</a> ↗ <a href="#">GitHub repository: Serverless</a> ↗
<i>Telemetry</i>	<code>telemetry</code>	The Telemetry module collects application logs and distributed traces for your application, and dispatches them to your preferred backends.	<a href="#">Telemetry Module [page 1921]</a> <a href="#">What's New for Telemetry</a> <a href="#">GitHub repository: Telemetry</a> ↗

## Other SAP Modules

These modules are developed and maintained by SAP teams outside of Kyma. To get help or request a feature, contact the module provider directly.

Other SAP Modules

Module	Technical name	Purpose	Documentation
<i>Transparent Proxy</i>	<code>transparent-proxy</code>	Use the transparent proxy for Kubernetes to connect workloads in a Kubernetes cluster to Internet and on-premise applications.	<a href="#">Transparent Proxy in the Kyma Environment</a>
<i>Connectivity Proxy</i>	<code>connectivity-proxy</code>	Use the connectivity proxy for Kubernetes to connect workloads in a Kubernetes cluster to on-premise systems, exposed via the Cloud Connector.	<a href="#">On-Premise Connectivity in the Kyma Environment</a>

### 2.3.3.3 Kyma Functionalities

SAP BTP, Kyma runtime and open source project "Kyma" offer slightly different functionalities and install a different set of components.

For all functionalities that the Kyma environment offers, see the official [project "Kyma" documentation](#).

Functionality Comparison

Functionality	open source project "Kyma"	SAP BTP, Kyma runtime
Service Level Agreements	✗	✓
Managed Kubernetes	✗	✓
Managed Kyma	✗	✓
Kyma CLI	✓	limited (SAP BTP, Kyma runtime supports commands for serverless Functions, not the commands related to installation.)
Centrally hosted Kyma dashboard	✗	✓
System landscape management in SAP BTP cockpit	✗	✓
In-cluster system landscape management (Application Connector)	✓	✗

### 2.3.3.4 Kyma Runtime: Basic Concepts

This table explains basic concepts relating to the Kyma environment. It aims to give you an understanding of the Kyma environment before you actually start using it to build extensions for your SAP solutions.

#### ⓘ Note

For an overview of the basic Kubernetes concepts that the Kyma environment heavily relies on, see the official [Kubernetes documentation](#).

Concept	Description
Kyma cluster	A Kubernetes cluster provisioned with the latest version of the open-source project "Kyma". You can enable such a cluster on a given subaccount through the SAP BTP cockpit. After creating a Kyma environment instance on your subaccount, the cluster is provisioned automatically through Gardener on your chosen cloud service provider. To access the cluster, you must have appropriate roles assigned to your subaccount.
Kyma module	An extension to the Kyma environment that can be added, deleted, or re-configured at runtime.
Role	Access to every cluster is managed by the roles assigned. Roles give the assigned users a different level of permissions suitable for different purposes. For more information, read <a href="#">Assign Roles in the Kyma Environment [page 3137]</a> .
Namespace	Namespaces are used to organize objects in a cluster and provide a way to divide cluster resources. This way, several users can share a cluster but have access only to resources within the namespace they have permissions for. This allows for increasing the security and organization of your cluster by dividing it into smaller units. Access to namespaces in the Kyma environment depends on your Kubernetes RBAC permissions.
Service operator	A service operator is a piece of software that provides a set of all necessary resources (such as CustomResourceDefinitions and controllers) needed to provision third-party services in your Kubernetes cluster.
Binding	The connection you create between a service instance and an SAP solution so that they can communicate with each other. You can also bind a service instance to any workload running in the Kyma environment, such as a Function or a microservice.
Credentials / Secrets	Sensitive data necessary for an SAP solution to call the service, connect to it, and authenticate it. Depending on whether you use Kyma dashboard or kubectl to create the binding between a service instance and an SAP solution, the Kubernetes Secret object that contains these credentials is either created automatically or you need to create it manually.
Function	A simple code snippet that you can run without provisioning or managing servers. It implements the exact business logic you define. A Function is based on the Function custom resource and can be written in either Node.js or Python. A Function can perform a business logic of its own. You can also bind it to an instance of a service and configure it to be triggered whenever it receives a particular event type from the service or a call is made to the service's API. Functions are executed only if they are triggered by an event or an API call.

Concept	Description
Microservice	An architectural variant for extensions or applications, where you separate the tasks into smaller pieces that interact with each other as loosely coupled, independently deployable units of code. A failing microservice should not cause your whole application to fail. Microservices are packed in a container that is always running; it's idling if there is no load. The microservice should always be reachable even when the Pods move around. Microservices typically communicate through APIs.

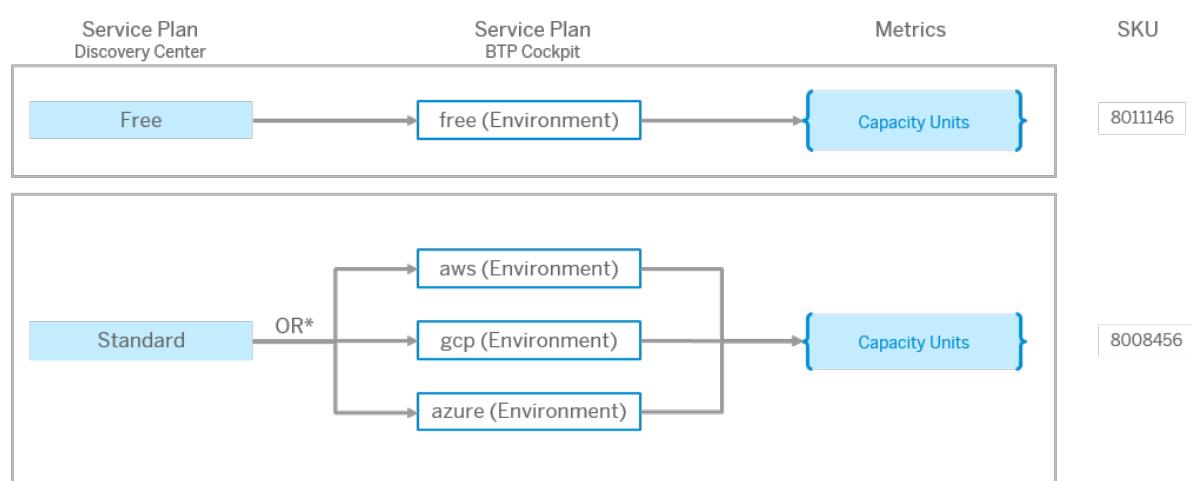
### 2.3.3.5 Service Plans and Metering for Kyma Runtime

This page explains the relationship between the service plans of the SAP Discovery Center and the service plans of the SAP BTP cockpit and provides information to help you understand how the service is billed.

## Service

### Overview

#### Consumption-Based Models



\* Depending on which hyperscaler the subaccount is hosted on

#### Subscription

No Subscription available for SAP BTP, Kyma runtime

#### SAP Discovery Center: Kyma Runtime

##### ⓘ Note

For more information, see [What is the Consumption-Based Commercial Model?](#)

## SAP BTP Cockpit: Service Plans

Name	Service Plan (Discovery Center)	Description
free	Free	Subscribe to the 30-day free plan provided on Amazon Web Services. This plan uses Kyma on a limited-size cluster (4 CPU—16 GB RAM). You can use the free plan only once in a global account for up to 30 days. The upgrade to the paid plan is not yet supported. Only best-effort support is available for free tier service plans, and these are not subject to SLAs. The services you plan to use must be available in the same region as the subaccount for the Kyma runtime.
aws	Standard	Select Amazon Web Services as the cloud provider where your Kyma cluster is deployed.
gcp	Standard	Select Google Cloud as the cloud provider where your Kyma cluster is deployed.
azure	Standard	Select Microsoft Azure as the cloud provider where your Kyma cluster is deployed.

## Metrics

The usage metric for the Cloud Service is a Capacity Unit (CU) per month.

Metric	Definition
Capacity Unit	Number of units consumed by the usage of the services as outlined in the solution-specific product supplement.  For SAP BTP, Kyma runtime, there are two capacity units: one to measure the workload, the second to measure the storage.

## Backward Calculation

### Formula

The relationship between consumed workload/storage and the respective CU is straightforward.

However, cost per monthly bill may vary because it depends on the size of the nodes that were used. There is no fixed formula because Kyma's flexible scaling.

### Underlying Metrics

In the following tables, "CU" stands for "Capacity Unit".

For CPU

Amount of CPU/CPU Nodes	Number of CU per Hour
2	0.4805555555555557 x 0.75
4	0.4805555555555557
8	0.4805555555555557 x 2
16	0.4805555555555557 x 4
32	0.4805555555555557 x 8
...	...

For Storage

Amount of Storage	Number of CU per Hour
1GB	0.00056423611
32GB	0.01805555552 (0.00056423611 x 32)

## Examples

- Your bill for Nodes shows 1800 capacity units charged for 4vCPU Nodes in a month.  
This means that you were running roughly  $1800 / 0.4806 = 3745.32$  hours worth of 4vCPU nodes. Given that a month has 720 hours, this means you were running  $3745.32/720 = 5.2$  Nodes cluster for a full month.  
Due to Kyma's automatic scaling, 5.2 does not mean actual size. It means that during some hours in the month, the cluster size was 4 nodes. At other times, there was more load, so the cluster autoscaled to 5 or 6 nodes, based on your configurations. So your average cluster size for the full month was 5.2 Nodes.
- Your bill for storage shows 200 capacity units charged for storage (32GB block) for the month.  
This means that you used  $200 / 0.0181 = 11049.72$  hours of 32GB storage blocks. Given that a month has 720 hours, this means that you have used  $11049.72/720 = 15.34$  blocks of 32GB over the month. So, you used  $15.34 * 32 = 491$ GB of storage continuously during the month.  
This implies that as you started using Kyma, you deployed more applications that used storage of various sizes, such as 4GB or 8GB.  
Note that storage is provided in blocks of 32GB, so if you used 33 GB, you would be charged for  $2 * 32$ GB, that is 64 GB.

## Calculator

Find the Kyma price calculator on <https://kyma-project.github.io/price-calculator/> and use it to estimate the costs for your SAP BTP, Kyma runtime.

1. Choose the **size of virtual machine** (VM).
2. Choose the **minimum of VMs** you need.
3. Estimate the **hours per month** you expect to run them.
4. Optionally, add more Nodes and more storage to the calculation.

Note that the result is an estimate, and the monthly bill may vary depending on the actual hours and size of the Kyma cluster (workload and storage) that was running in a month.

## **Supplemental Terms And Conditions**

For more information, see [SAP Business Technology Platform Service Description Guide](#), section “SAP BTP, KYMA RUNTIME”.

## **Glossary**

[Commercial Information Glossary](#)

## **Service Specifics**

As soon as you instantiate a Kyma cluster, the basic costs for the empty cluster incurs.

Cost may then vary depending on the actual workloads and storage you consume per month.

### **2.3.4 Neo Environment**

The Neo environment lets you develop HTML5, Java, and SAP HANA extended application services (SAP HANA XS) applications. You can also use the UI Development Toolkit for HTML5 (SAPUI5) to develop rich user interfaces for modern web-based business applications.

See also:

- [What Is SAP BTP, Neo Environment](#)
- [Development, Neo Environment](#)
- [Extensions, Neo Environment](#)
- [Administration and Operations, Neo Environment](#)
- [Security, Neo Environment](#)
- [Getting Support, Neo Environment](#)

### **Migration from the Neo to the Multi-Cloud Foundation**

To learn more about why and how to migrate your scenarios on SAP BTP, see [Migrating from the SAP BTP Neo Environment to the Multi-Cloud Foundation](#).

## 2.4 Trial Accounts and Free Tier

Explore the different options for trying out SAP BTP.

### Trial Account or Free Tier Offering: When to Use Which?

Before setting up your account, you need to decide which free offering for SAP BTP is suitable for your needs:

- **Pay-As-You-Go or CPEA accounts with free tier service plans** are open to customers, partners, and let you try out SAP BTP in a per service defined time span for free. These account types enable you to test your scenarios and generally offer the option to upgrade to paid service plans. These accounts also allow you to store data long-term and move projects to production. You also get access to our community, including free technical resources such as tutorials and blog posts. For more information, see [Enterprise Accounts \[page 84\]](#) and [Using Free Service Plans \[page 91\]](#).

#### ⓘ Note

When using Kyma, the free plan is available only once in a global account for up to 30 days. For more information, see [Available Plans in the Kyma Environment](#).

You can self-register for an enterprise account with free tier service plans. For more information, see [Get an Account on SAP BTP to Try Out Free Tier Service Plans](#).

#### ⓘ Note

Only community support is available for free tier service plans and these are not subject to SLAs.

#### ⓘ Note

The option to upgrade from free tier service plans to paid service plans is not yet available for all services and runtimes, such as Kyma. See [Using Free Service Plans](#) to read more about upgrading from free tier service plans to paid service plans.

- A **trial account** lets you try out SAP BTP for free for 90 days. The services provided for the trial account allow restricted use of the platform resources and services. Access is open to everyone. Trial accounts are intended for personal exploration, your own non-productive testing, and evaluation of the services in accordance with [SAP BTP trial terms and conditions](#). A SAP BTP trial account must not be used for production use or team development. You are not permitted to use the trial account in any productive or commercial manner.

You can self-register for a trial account. For more information on how to do that, see [Get a Free Account on SAP BTP Trial](#).

### Free Tier and Always Free Tags in the SAP Discovery Center

In the [SAP Discovery Center service catalog overview](#), some services display the tag *Free Tier*, which indicates the service is offering a free tier service plan. Some services display the tag *Always Free*, which

indicates the service is offering a service plan that comes free of additional charges, as it is already included in your overall SAP BTP contract. You can use the filter function of the SAP Discovery Center to filter for services that offer *Free Tier* and *Always Free* plans. The *Always Free* service plans include the following note in their service plan description: "This service plan is included in the overall SAP Business Technology Platform contract".

### Note

*Always Free* service plans might not be available in all regions or for all providers.

## Trial Lifecycle

- **Use your SAP BTP trial account:**

Familiarize yourself with the [Trial Scope \[page 83\]](#) or try out one of our [Starter Scenarios](#)  on the [Tutorial Navigator](#) .

- **Extend your SAP BTP trial account in intervals:**

- Your 90-day trial period is divided in intervals. If you sign in to your trial account regularly, the intervals are extended automatically for you by up to 30 days or until your overall 90-day trial is finished.
- If you don't sign in to your trial account for 30 days or more, your account will be suspended. All suspended accounts will be deleted and you won't be able to use applications or services.

- **Delete your SAP BTP trial account:**

After 90 days, your trial account is automatically deleted. But if you want to proactively delete your SAP BTP trial account, you can navigate to the global accounts scope and select the Account Explorer page, then click the [Delete Trial Account](#) button.

- If you want to continue to use an SAP BTP trial account after deletion, you need to set up a new account.
- If you want to explore SAP BTP without time limit, create an enterprise account with free tier service plans allowing you to test your scenarios. See: [Get an Account on SAP BTP to Try Out Free Tier Service Plans](#) .

### Caution

Environments enabled on your trial account may not be valid for your whole trial period. To learn about the validity period for the Kyma environment, see [About the Trial Account \[page 227\]](#).

## Trial Scope

- A trial account enables you to explore the basic functionality of SAP BTP.
- SAP BTP trial accounts are available in several regions. For more information, see [Regions \[page 17\]](#).
- You can create directories in your trial account. For more information, see [Managing Directories Using the Cockpit \[page 2165\]](#)
- You can use productive and beta services. To consume beta services, you must enable the subaccount for beta features during the subaccount creation or when you edit the subaccount details.

- You can manage platform users by assigning them role collections. For more information, see [Working with Role Collections](#).
- A trial account includes 4 GB of memory for applications.
- You can use 8 GB of instance memory.
- There are 10 total routes and 40 total services available.
- You can use 2 configured on-premise systems with the Cloud connector.
- There's no service level agreement with regards to the availability of the platform.
- You can use HDI containers in a shared SAP HANA database (only available on cf-us10).
- For cleanup purposes, applications stop automatically on a daily basis. You need to manually restart them when needed.

 **Note**

Applications are stopped at midnight (or some time later depending on server load) relative to the region in which you created your trial account. If you're working in a time zone that is far from the region where your trial account was created, then your applications may stop during business hours.

- You can have a trial tenant of SAP Cloud identity Services, see [Get Your Tenant](#).

## Related Information

[Getting Started with a Trial Account in the Cloud Foundry Environment \[page 147\]](#)

[Getting Started with a Trial Account in the Kyma Environment \[page 226\]](#)

## 2.5 Enterprise Accounts

Enterprise accounts are usually associated with SAP customer or partner contracts and contain their purchased entitlements to platform resources and services. However, it's also possible to create an enterprise account for personal exploration.

SAP BTP provides different types of global accounts, **enterprise** and **trial**. The type you choose determines pricing, conditions of use, resources, available services, and hosts.

If you want to start out using services for free, you can create an enterprise account and only use free tier service plans: The major benefit of using such an account for exploration purposes is that you can upgrade your services to productive use without losing any of your data. Check out the [Free Tier Services](#) in SAP Discovery Center and learn the details about [Using Free Service Plans \[page 91\]](#) in your enterprise account.

Another way to start out using services for free is to create a trial account. But keep in mind that when you want to use your global account productively, there's no way to migrate your data from the trial account to a productive enterprise account. For more information, see [Trial Accounts and Free Tier \[page 82\]](#).

## Enterprise Accounts

The main features of enterprise accounts are described for use by customers and by partners in the following table:

	Customer Account	Partner Account
<b>Use case</b>	<p>A global account that enables you to host productive, business-critical applications with 24/7 support.</p> <p>You can purchase a global account just like any other SAP software. You can upgrade and refine your resources later on. You can also contact your SAP sales representative and opt for a configuration, tailored to your needs.</p> <p>If you want to start out exploring services for free, you can sign up for a Pay-As-You-Go for SAP BTP global account and make use of the free tier services only. See <a href="#">Using Free Service Plans [page 91]</a>.</p>	<p>A global account that enables you to build applications and to sell them to your customers.</p> <p>If you want to start out exploring services for free, you can sign up for a Pay-As-You-Go for SAP BTP global account and make use of the free tier services only. See <a href="#">Using Free Service Plans [page 91]</a>.</p>
<b>Benefits</b>	Support for productive applications.	<ul style="list-style-type: none"><li>Includes SAP Application Development licenses that enable you to get started with scenarios across cloud and on-premise applications.</li><li>Offers the opportunity to certify applications and receive SAP partner logo package with usage policies.</li><li>Advertise and sell applications via the SAP Store</li></ul>
<b>Services available</b>	Productive services.	Productive and beta services.
<b>Limitations</b>	Resources according to your contract.	Predefined resources according to your partner package. You can purchase additional resources if necessary.

	Customer Account	Partner Account
<b>Registration</b>	<p>For more information about SAP Integration Suite and SAP Extension Suite pricing, see <a href="https://www.sap.com/products/extension-suite/pricing.html">https://www.sap.com/products/extension-suite/pricing.html</a> and <a href="https://www.sap.com/products/integration-suite/pricing.html">https://www.sap.com/products/integration-suite/pricing.html</a>.</p> <p>Contact us on <a href="#">SAP BTP</a> or via an SAP sales representative.</p> <p>If you're located in China and want to buy a global account on SAP BTP you need to contact an SAP sales representative: <a href="https://www.sap.cn/registration/contact.html">https://www.sap.cn/registration/contact.html</a>.</p>	Sign up to join the <a href="#">SAP Partner Program</a> .
<b>Available Regions</b>	See <a href="#">Regions [page 17]</a> .	See <a href="#">Regions [page 17]</a> .

## Cancelling a SAP BTP Enterprise Account

As a business user, you can cancel your contract via your account executive. As an individual user with a trial account, you can cancel your account in your [SAP for Me account](#). You can also contact the SAP store via their "Contact Us" function.

For the commercial models available for enterprise accounts, see [Commercial Models \[page 86\]](#).

## Related Information

[Getting Started in the Kyma Environment \[page 223\]](#)

[Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#)

[Getting Started with an Enterprise Account in the Cloud Foundry Environment \[page 158\]](#)

## 2.5.1 Commercial Models

SAP BTP offers two different commercial models for enterprise accounts.

- **Consumption-based commercial model:** Your organization receives access to all current and future services that are eligible for this model. You have complete flexibility to turn services on and off and to switch between services as your business requires throughout the duration of your contract. This commercial model is available in the following flavors: SAP BTP Enterprise Agreement (SAP BTPEA), Cloud Platform Enterprise Agreement (CPEA), and Pay-As-You-Go for SAP BTP.

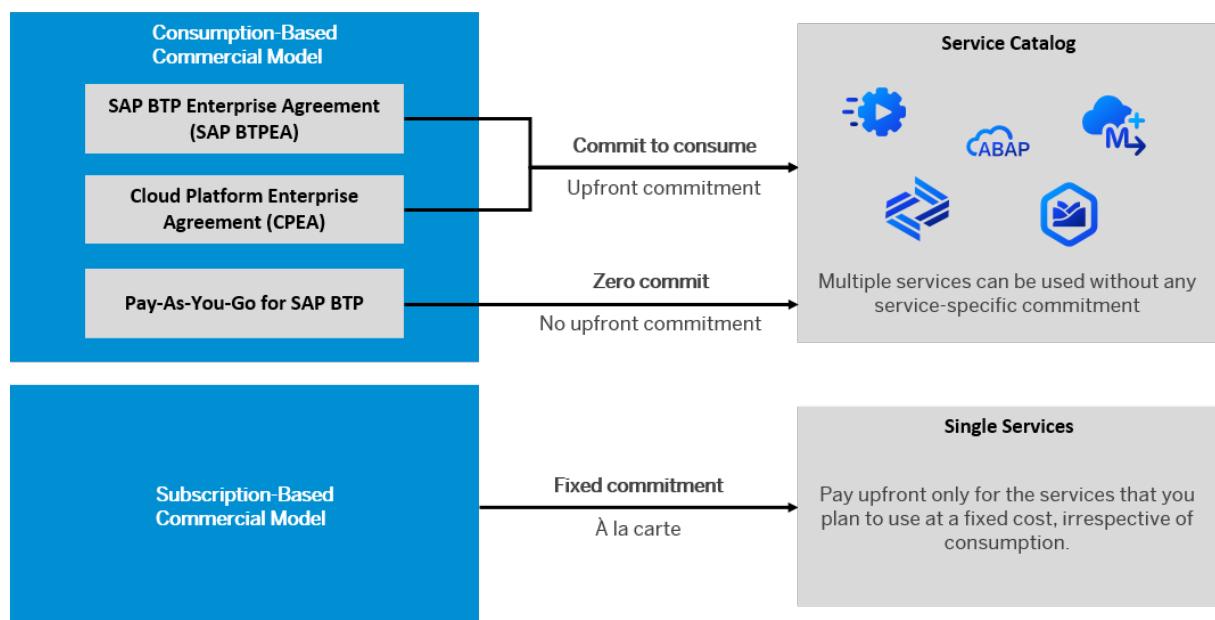
For more information, see [What Is the Consumption-Based Commercial Model? \[page 87\]](#)

- **Subscription-based commercial model:** Your organization subscribes only to the services that you plan to use. You can then use these services at a fixed cost, irrespective of consumption.
- For more information, see [What Is the Subscription-Based Commercial Model? \[page 90\]](#)

#### ⓘ Note

You can use both commercial models, either in separate global accounts or in the same global account depending on your business needs. Contact your SAP account executive or sales representative for more information. Note however that you cannot mix more than one flavor of the consumption-based commercial model in the same global account.

For information about service availability, prices, and estimators, see <https://www.sap.com/products/technology-platform/solutions.html> and <https://www.sap.com/products/technology-platform/integration-suite/pricing.html>. You can also view the service catalog via the [SAP Discovery Center](#).



### 2.5.1.1 What Is the Consumption-Based Commercial Model?

#### ⓘ Note

The use of the consumption-based commercial model is subject to its availability in your country or region.

With the consumption-based model, your organization purchases an entitlement to all current and future SAP BTP services that are eligible for this model. Throughout the duration of your contract, you have complete flexibility to turn services on and off and to switch between services as your business requires.

The consumption-based commercial model is available in the following flavors: the **SAP BTPEA (SAP BTP Enterprise Agreement)**, the **CPEA (Cloud Platform Enterprise Agreement)**, and **Pay-As-You-Go for SAP BTP**. Each option is suited to different business situations and levels of financial commitment, as described in the table below. For additional information and clarifications, please contact your account executive.

## SAP BTPEA

- Your organization makes a prepaid investment in cloud credits for the contract duration with an annual commitment to consume SAP BTP services.
- This model is suitable for customers that have well-established and planned use cases, and who want the flexibility of turning services on and off, and switching between services, without the commitment of being tied to a single service throughout the duration of the contract.
- You receive a monthly balance statement that provides information about the usage consumption of each service and the corresponding costs. The total monthly cost is deducted from your cloud credits balance.
- This model has a minimum investment entry, and volume-based discounts are available.
- You are billed annually in advance. Any overages are billed in arrears at list price.
- You can top up your cloud credits at any time to prevent overages.
- You get access to new SAP BTP services that are added to the consumption-based service catalog.

---

## CPEA

### ⓘ Note

Certain restrictions apply. See the *Frequently Asked Questions* section in <https://www.sap.com/products/technology-platform/pricing.html>.

- Your organization makes a prepaid investment in cloud credits for the contract duration with an annual commitment to consume SAP BTP services.
  - This model is suitable for customers that have well-established and planned use cases, and who want the flexibility of turning services on and off, and switching between services, without the commitment of being tied to a single service throughout the duration of the contract.
  - You receive a monthly balance statement that provides information about the usage consumption of each service and the corresponding costs. The total monthly cost is deducted from your cloud credits balance.
  - This model has a minimum investment entry, and volume-based discounts are available.
  - You are billed annually in advance. Any overages are billed in arrears at list price.
  - You can top up your cloud credits at any time to prevent overages.
-

## Pay-As-You-Go for SAP BTP

- You have the same access to all the services that are available in SAP BTPEA, but with a highly flexible zero-commitment model – you pay nothing upfront and there is no minimum usage requirement or annual commitment.
- You pay only for the SAP BTP services that you want, when you use them.
- You are billed monthly in arrears.
- Service charges are non-discountable.
- This low-risk model is suitable for customers with use cases that are not well defined, and are interested in running a proof-of-concept in a productive environment. This model provides the flexibility of turning services on and off, and switching between services, as needed throughout the duration of the contract.
- A seamless transition to the CPEA model is available, on the condition that you have no other CPEA-based global accounts.

If you want information about the **Pay-As-You-Go for SAP BTP for cloud test, demo, and development** commercial model , go to the [SAP Partner Portal](#).

### ⓘ Note

- In global accounts that use the consumption-based commercial model, SAP BTP, Cloud Foundry runtime is not listed in the [Entitlements](#) pages in the SAP BTP cockpit. A technical limit of 200 GB of Cloud Foundry runtime memory is assigned by default to every subaccount. The limit defines the maximum amount of runtime memory that can be used in the subaccount. Note that the “Standard” plan in the consumption-based commercial model is a paid plan and that you are billed based on the amount of Cloud Foundry runtime you consume. For more information on consumption monitoring, see [Consumption Monitoring](#).
- If you need to increase this limit, report an incident to [SAP support](#) on the BC-NEO-CIS component. This also applies to other services that have a technical quota limit.

### → Tip

You can monitor costs and service usage throughout the contract period. See [Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#).

For information about eligible services and pricing, see <https://www.sap.com/products/technology-platform/pricing.html>, or access the SAP BTP service catalog via the [SAP Discovery Center](#). The SAP BTP service catalog allows you to identify service availability per data center and to determine licensing model compatibility per available service plan.

As part of the free tier model for SAP BTP, some services offer free service plans that allow you to try out these services without any additional charge within your SAP BTPEA, CPEA, or Pay-As-You-Go for SAP BTP global account. For more information, see [Using Free Service Plans \[page 91\]](#).

To find frequently asked questions about the consumption-based commercial model, see the [Frequently Asked Questions](#) section in <https://www.sap.com/products/technology-platform/pricing.html>.

### Note

Depending on your business needs, you can combine the consumption-based commercial model with the subscription-based commercial model in the same global account, or you can use both commercial models in the same global account. Contact your SAP account executive or sales representative for more information. Note however that you cannot mix more than one flavor of the consumption-based commercial model in the same global account.

## Related Information

[What Is the Subscription-Based Commercial Model? \[page 90\]](#)

[Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#)

[View Subaccount Usage Analytics \[page 2182\]](#)

[Managing Global Accounts Using the Cockpit \[page 2143\]](#)

[Managing Directories Using the Cockpit \[page 2165\]](#)

[Managing Subaccounts Using the Cockpit \[page 2173\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

### 2.5.1.2 What Is the Subscription-Based Commercial Model?

Your organization receives a fixed price and period (typically a 1 to 3-year period) for access to your subscribed SAP BTP services.

Under this commercial agreement:

- You are entitled to use only the subscribed services.
- To access additional services, at an extra cost, you can modify your contract via your sales representative or account executive.
- You pay at a fixed cost, regardless of consumption of subscribed services.
- You pay in advance when the contract period starts.
- Your organization can renew the subscription at the end of the contract period.

For information about available services and pricing, see [SAP Store](#).

You can also access the SAP BTP service catalog via the [SAP Discovery Center](#) to identify the availability of services by data center, and also to determine licensing model compatibility per service plan.

Note that some services can be subscribed based on a user metric or a resource metric. For example, a Portal service can be based on the number of site visits or user metrics. A resource-based metric is more common when dealing with a large number of users; for example, suppliers accessing a portal to interact with your organization. Since it isn't always possible to predict how many resources would be required upfront for a three-year period, you can increase your original order if resource usage exceeds your subscribed quota. Using SAP BTP cockpit, you can view resource consumption within your global account on a monthly basis.

In the subscription-based model, you also get access to bundles or packages that comprise several related services and apps. Most of the time, this works out to be more cost effective when compared to subscribing to individual SAP BTP services.

## Can an existing global account under a consumption-based contract be transformed to the subscription-based model?

You can change your existing contract from the consumption-based commercial model to a subscription license. Keep in mind that not all services that are eligible for the consumption-based model are compatible with the subscription-based model. We recommend that you contact your SAP account executive or sales representative to discuss feasibility and terms of transforming your contract.

Note that you can use both consumption and subscription-based commercial models either in separate global accounts or in the same global account depending on your business needs. Contact your SAP account executive or sales representative for more information.

## Related Information

[What Is the Consumption-Based Commercial Model? \[page 87\]](#)

[Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#)

[View Subaccount Usage Analytics \[page 2182\]](#)

[Managing Global Accounts Using the Cockpit \[page 2143\]](#)

[Managing Directories Using the Cockpit \[page 2165\]](#)

[Managing Subaccounts Using the Cockpit \[page 2173\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

## 2.5.2 Using Free Service Plans

The free tier model for SAP BTP lets you try out services in global accounts without any additional cost using the consumption-based commercial model and an [enterprise account](#).

We offer free service plans for many services within SAP BTP. Free service plans are limited in scope and capacity. They are designed to allow you and your team to explore new SAP BTP capabilities before committing to subscribing to a full capacity paid service.

The free tier model for SAP BTP is added automatically to new and existing contracts using the consumption-based commercial model. There, the free service plans are visible.

### Note

The use of the consumption-based commercial model is subject to its availability in your country or region.

The consumption-based commercial model is available in the following flavors: the **SAP BTPEA** (SAP BTP Enterprise Agreement), the **CPEA (Cloud Platform Enterprise Agreement)** and **Pay-As-You-Go for SAP BTP**. For more information, see [What Is the Consumption-Based Commercial Model? \[page 87\]](#).

To try out services that participate in the free tier model for SAP BTP, create a service instance using the **Free** service plan offered by the service.

#### → Tip

To find out which services offer the free plans, see [Discover Free Services \[page 92\]](#). Here, you also find information on how to identify the technical limitations of the free plan of each service. [Access Free Services \[page 93\]](#) by assigning entitlements to subaccounts and creating service instances.

Once you reach the technical limits of a service and want to continue using it, you must move to a paid service plan. Upgrading an existing service instance from free tier to paid usually doesn't require creating a new service instance. Find out how to upgrade a service plan at [Upgrade to a Paid Service Plan \[page 93\]](#).

Use of free tier service plans is subject to additional terms and conditions as provided in the [Business Technology Platform Supplemental Terms and Conditions](#). Only community support is available for Free service plans and these aren't subject to SLAs.

## Discover Free Services

Find out which services offer Free plans and what the limitations are.

Free Tier services can be explored in the service catalog of SAP Discovery Center. A free tier service appears as a "Free" plan offering for a service.

1. Access the [Service Catalog](#) in the [SAP Discovery Center](#).
2. In the [Categories](#) section, select [Free Tier Services](#).

The screenshot shows the SAP Discovery Center web interface. At the top, there's a dark header bar with the SAP logo and navigation links for 'Home' and 'Services'. Below the header is a search bar and a 'Services' button. On the left, a sidebar titled 'Categories' has a tree view where 'Free Tier Services' is selected. The main area displays a grid of service cards. One card in the second row, third column is explicitly labeled 'FREE TIER'. Each service card contains a small icon, a title, and some descriptive text. To the right of the grid, there are filter and sort options.

3. Select one of the services displayed in the catalog.
4. Navigate to the [Pricing](#) tab.

On this page, you can see all available plans for this service, including the limitations of each plan. Plans that participate in the free tier model for SAP BTP are labeled **Free**.

## Access Free Service Plans

To try out the free tier model for SAP BTP, you must entitle your subaccount for the free plan and create a service instance using this plan.

### ⓘ Note

The free tier model for SAP BTP is available for accounts that use the consumption-based commercial model.

1. Enter your global account.  
For more information, see [Access the Cockpit](#).
2. Configure entitlements for the subaccount in which you want to use the free service plans. Select the service and choose the free plan.  
For more information, see [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).
3. Navigate into the subaccount and go to the [Service Marketplace](#).
4. Find the service you want to try out and select it.  
You find a description of the available service plans, including the free service plan. By clicking [More](#), you find the scope and limitations of the free plan for this service.
5. Create an instance of the service and select the free service plan.  
For more information, see [View and Manage Services from the Service Marketplace](#).

You can now start using this service.

## Update to a Paid Service Plan

Once you've reached the limits of the free plan, you can upgrade the free service plan to a paid plan.

### ⓘ Note

The option to upgrade from free tier service plans to paid service plans is not yet available for all services and runtimes, such as Kyma. To use a paid service plan for Kyma runtime, you will need a new Kyma cluster. It is possible to have both - free and paid clusters - in parallel.

### ⓘ Note

To update the ABAP environment to a paid service plan, you first need to configure entitlements and quotas for the ABAP environment. You need at least the following entitlements assigned to your subaccount:

- service plan: [standard](#)
- quota plans:
  - [1 abap\\_compute\\_unit](#)
  - [2 hana\\_compute\\_unit](#)

For more information, see [Increasing the Quota for the ABAP Environment \[page 177\]](#).

### Note

It's not possible to switch from a paid plan to a free plan.

1. In your subaccount, navigate to the [Instances and Subscriptions](#) page.
2. Find the instance of the service and click  (*Actions*) and [Update](#).
3. In the [Update Instance](#) dialog, change the plan to a paid service plan.
4. Click [Update Instance](#) to save your changes.

You can now use the service without the limitations of the free service plan.

## Scope

- The free tier model for SAP BTP is available for CPEA and Pay-As-You-Go for SAP BTP contracts. For more information, see [What Is the Consumption-Based Commercial Model? \[page 87\]](#).
- Services using the free tier service plans run on the same platform as paid services.
- Services are technically limited. This limit depends on the service. You can find a description of the service plans in the [Service Marketplace](#) in the cockpit or in the [SAP Discovery Center](#).

## 2.6 Account Model

Learn more about the different types of accounts on SAP BTP and how they relate to each other.

Accounts are structured according to global accounts, subaccounts, and directories.

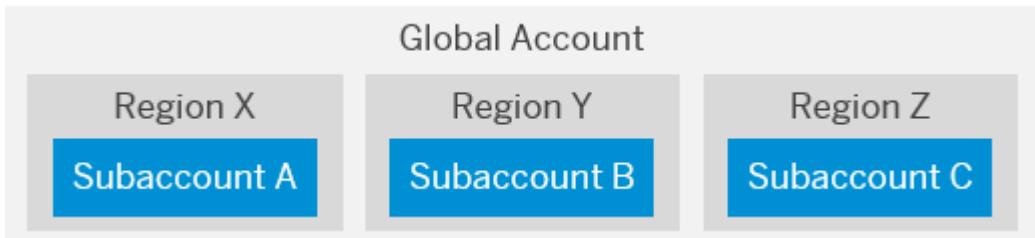
To learn more about managing your account model, see [Account Administration \[page 2139\]](#).

### 2.6.1 Global Accounts

A **global account** is the realization of a contract you or your company has made with SAP.

A global account is used to manage subaccounts, members, entitlements and quotas. You receive entitlements and quotas to use platform resources per global account and then distribute the entitlements and quotas to the subaccount for actual consumption. There are two types of commercial models for global accounts: consumption-based model and subscription-based model. See [Commercial Models \[page 86\]](#).

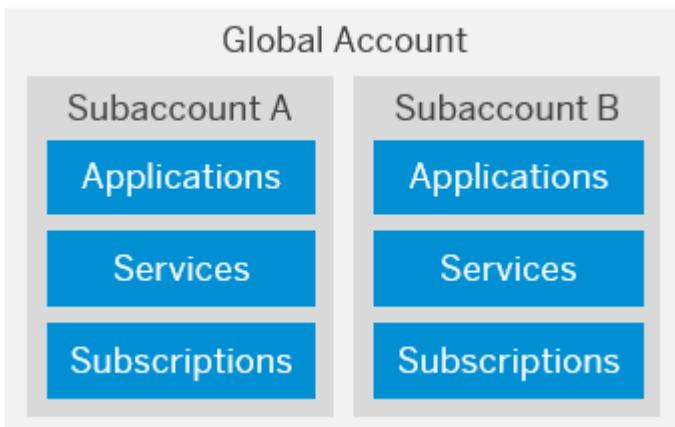
Global accounts are region- and environment-independent. Within a global account, you manage all of your subaccounts, which in turn are specific to one region.



## 2.6.2 Subaccounts

**Subaccounts** let you structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements.

A global account can contain one or more subaccounts in which you deploy applications, use services, and manage your subscriptions. Subaccounts in a global account are independent from each other. This is important to consider with respect to security, member management, data management, data migration, integration, and so on, when you plan your landscape and overall architecture.



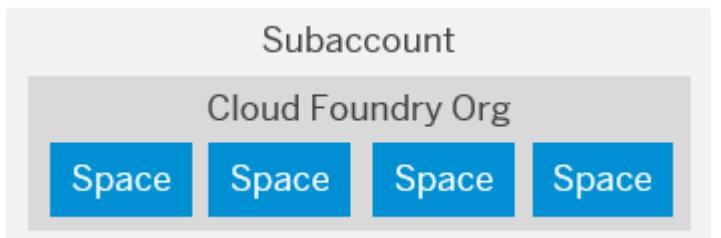
Each subaccount is associated with a region, which is the physical location where applications, data, or services are hosted. The specific region is relevant when you deploy applications and access the SAP BTP cockpit using the corresponding cockpit URL. The region assigned to your subaccount doesn't have to be directly related to your location. You could be located in the United States, for example, but operate your subaccount in Europe.

The entitlements and quotas that have been purchased for a global account have to be assigned to the individual subaccounts.

Global accounts and subaccounts are completely independent of user accounts. For more information, see [User and Member Management \[page 104\]](#).

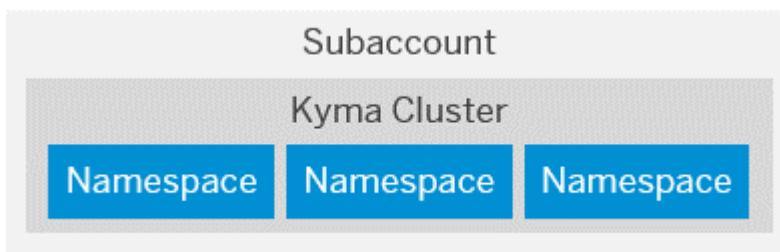
### Subaccounts, Cloud Foundry Orgs and Kyma Clusters

- When you enable the Cloud Foundry environment in one of your subaccounts, the system automatically creates a Cloud Foundry org for you. The subaccount and the org have a 1:1 relationship and the same navigation level in the cockpit (even though they may have different names). You can create spaces within that Cloud Foundry org. Spaces let you further break down your account model and use services and functions in the Cloud Foundry environment.



For more information about Cloud Foundry orgs and spaces, see the Cloud Foundry documentation at <https://docs.cloudfoundry.org/concepts/roles.html>.

- When you enable the Kyma environment in one of your subaccounts, the system automatically creates a Kubernetes cluster equipped with the latest version of the open-source project "Kyma" for you. The subaccount and the Kyma cluster have a 1:1 relationship. You can create namespaces within that Kyma cluster. Namespaces let you further break down your account model and use services and functions in the Kyma environment. For more information, see the Kubernetes documentation at <https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/>.

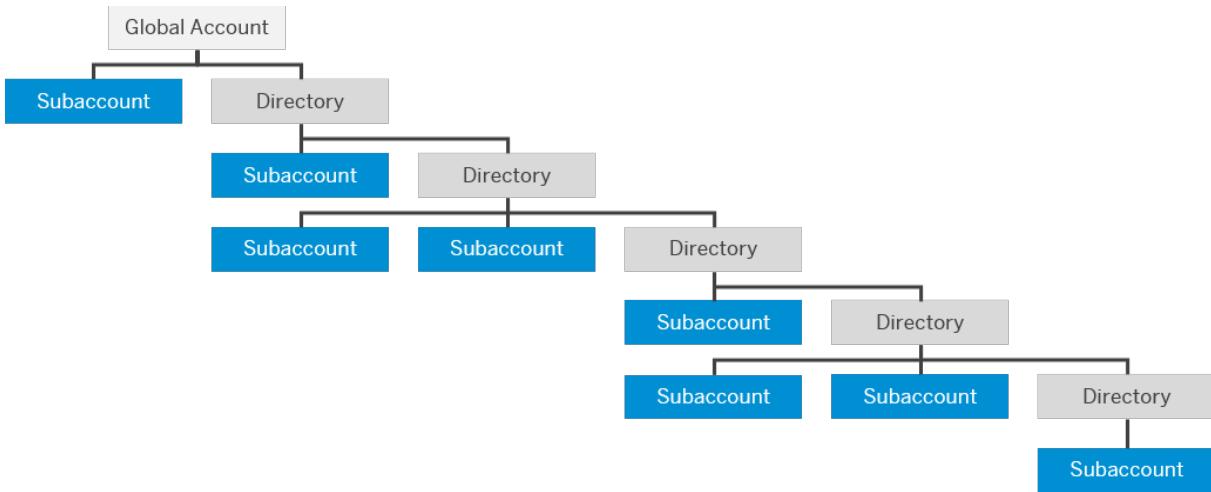


## 2.6.3 Directories

With **directories**, you can organize and manage your subaccounts according to your technical and business needs.

A directory can contain directories and subaccounts to create a hierarchy. Using directories to group other directories and subaccounts is optional - you can still create subaccounts directly under your global account.

You can create a hierarchical structure that is up to 7 levels deep. The highest level of a given path is always the global account and the lowest is a subaccount, which means that you can have up to 5 levels of directories between the global account and the lowest level subaccount.



Directories enable you to:

- Group and filter directories and subaccounts
- Monitor usage and costs for contracts that use the consumption-based commercial model

Optionally, you can also enable the following features in your directories:

- Manage Entitlements: Enables the assignment of a quota for services and applications to the directory from the global account quota, for distribution to the directory's subaccounts. When you assign entitlements to a directory, you express the entitlements and maximum quota that can be distributed across its children subaccounts. You also have the option to choose the auto-assignment of a set amount of quota to all subaccounts created or moved to that directory. Subaccounts that are already in the directory when you select that option will not be auto-assigned quota.

#### i Note

If you've enabled the Manage Entitlements feature for a given directory, you must first assign the necessary entitlements and maximum allowed quota from the global account to that directory. Then you can distribute this "reserved" quota to any of the directory's child subaccounts.

- Manage Authorizations: Enables authorization management for the directory. For example, it allows certain users to manage directory entitlements. You can only use this feature in combination with the [Manage Entitlements](#) feature.

## Related Information

[Getting a Global Account \[page 144\]](#)

[Setting Up Your Account Model](#)

[Managing Global Accounts Using the Cockpit \[page 2143\]](#)

[Managing Directories Using the Cockpit \[page 2165\]](#)

[Managing Subaccounts Using the Cockpit \[page 2173\]](#)

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Account Administration Using APIs of the SAP Cloud Management Service \[page 2378\]](#)

[Manage the Account Explorer Hierarchy \[page 2168\]](#)

[Configure Entitlements and Quotas for Directories \[page 2188\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

## 2.6.4 Relationship Between Global Accounts, Subaccounts, and Directories

A global account can group together different directories and subaccounts that an administrator makes available to users. Administrators can assign the available entitlements and quotas of a global account to its different subaccounts and move it between subaccounts that belong to the same global account.

The hierarchical structure of global accounts, directories, and subaccounts lets you define an account model that accurately fits your business and development needs. For example, if you want to separate development, testing, and productive usage for different departments in your organization, you can create a directory for each department, and within each directory, you group subaccounts for development, testing, and production.

## 2.6.5 Labels

Labels are user-defined words or phrases that you can assign to various entities in SAP BTP to categorize them in your global account, to identify them more easily.

For example, in the [Account Explorer](#) page in the cockpit, you can quickly filter for directories and subaccounts by label.

You can assign labels to these entities when you create or edit them using the SAP BTP cockpit, command line interface (btp CLI), or REST APIs:

- Directories
- Subaccounts
- Multitenant application subscriptions
- Service instances
- Environment instances

### ⓘ Note

For environment instances, these custom labels are user defined and apply only to SAP BTP. They are not the same labels that might be defined by your environment broker.

Labels are made up of a [label name](#) (also referred to as a [key](#)) and up to 10 [values](#) associated with the label. You can apply label names and values in any way that suits your business and technical needs.

Types of labels and examples

### Types of Labels

Single-value labels are useful for labels that have an identifier, or for labels with fixed lists.

### Examples

- Label Name: **Cost Object**  
Value: The ID of a cost center number or internal order that is associated with the entity, such as:  
**000001134789**
- Label Name: **Status**  
Value: **Active** or **Inactive**
- Label Name: **Landscape**  
Value: **Dev** or **Test** or **Production**

Multi-value labels make them useful for labels that typically have more than one value assigned to them.

- Label Name: **Contacts**  
Value: The e-mail addresses or names of users that are responsible for the entity, such as: **admin-1@example.com** and **admin-2@example.com**

Labels can also be assigned with a name only and no value. In such cases, the label behaves like tag.

- Label Name: **For demo only**
- Label Name: **Audited**
- Label Name: **Flagged for deletion**

Note that currently for service instances, labels must have at least value.

### ⓘ Note

Labels replace what were previously called "custom properties". Custom properties supported only single values per label and were available only to directories and subaccounts. As a result of the move to labels, all relevant commands in the SAP BTP command line interface (btp CLI) and in the relevant REST APIs in the SAP Cloud Management service have been updated accordingly. The `custom-properties` parameter in the btp CLI and the `customProperties` field in the relevant REST APIs are deprecated.

You can assign labels when you create an entity, and then later add, change, or remove labels by editing the entity.

### → Tip

- In the [Account Explorer](#) and [Instances and Subscriptions](#) pages in the SAP BTP cockpit, assigned labels are shown in the **Labels** column. To display the column if it is not shown, click ([Configure Table Columns](#)).
- In the [Account Explorer](#) page, you can view the labels that are assigned to a directory or subaccount by choosing the [More Info](#) option of each directory and subaccount. Assigned labels are also listed under the **Labels** tab when you display the [Overview](#) page of every directory and subaccount.
- In the [Account Explorer](#) and [Instances and Subscriptions](#) pages, you can filter the displayed entities by their assigned labels in the **Search** field.
- In the [Instances and Subscriptions](#) page, you can also view the labels that are assigned to a subscription or instance by expanding its details panel.

When working with labels, consider the following aspects:

- In the cockpit, each entity can have up to 10 labels assigned to it.
  - You cannot add the same label name more than once to the same entity.
  - Existing label names and values are offered as suggestions when you or anyone else assigns a label to other entities of the same type in your global account.
- When you view a subaccount in the cockpit, the subaccount also shows the labels that are assigned to its parent directory and to other directories that are above it in the same path in your account structure. And in the *Account Explorer*, when you filter by labels that are assigned to a directory, the subaccounts in that directory path are also listed. We refer to these as inherited labels.
- Label names and values are case-sensitive, which means you can create variants of the same label name with a different casing; for example, **My Label** and **My label** can coexist as separate labels. We recommend that you avoid using different casing or styling to create variants of the same names or values.

## Related Links

- [Create a Subaccount \[page 2173\]](#)
- [Change Subaccount Details \[page 2180\]](#)
- [Create a Directory \[page 2166\]](#)
- [Cloud Management Tools — Feature Set Overview \[page 129\]](#)
- [Commands in the btp CLI \[page 2354\]](#)
- [Account Administration Using APIs of the SAP Cloud Management Service \[page 2378\]](#)
- [Managing Service Resources Using the APIs of the SAP Service Manager \[page 2420\]](#)
- [Creating Service Instances](#)
- [Working with Environment Instances](#)
- [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#)

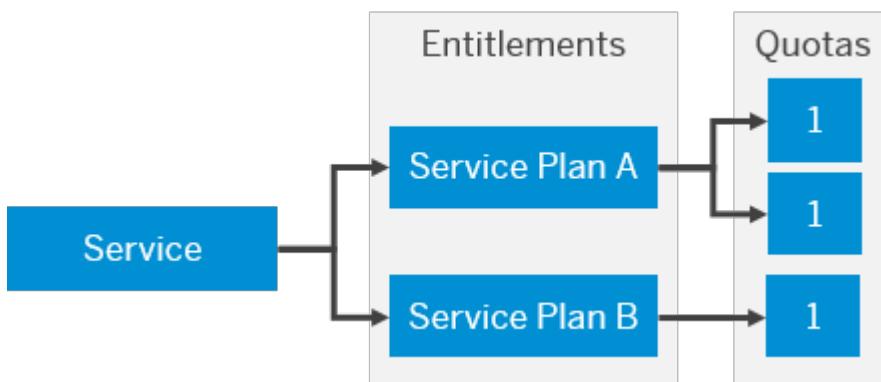
## 2.7 Entitlements and Quotas

When you purchase an enterprise account, you're entitled to use a specific set of resources, such as the amount of memory that can be allocated to your applications.

## Services and Service Plans

On SAP BTP, all external dependencies such as databases, messaging systems, file systems, and so on, are **services**. In this context, multitenant applications and environments are considered services.

Each service has one or more **service plans** available. A service plan is the representation of the costs and benefits for a given variant of a particular service. For instance, a database may be configured with various "T-shirt sizes", each of which is a different service plan.



## Entitlements

An **entitlement** is your right to provision and consume a resource. In other words, entitlements are **the service plans** that you're entitled to use.

## Quotas

A **quota** represents the numeric quantity of a service plan that you're entitled to consume in your global account and its subaccounts.

From the perspective of your SAP BTP global account contract, the global quota of a service plan can be defined as either fixed or unlimited:

- **Fixed quota:** The quota or allowance of the service plan is the upper limit at which the plan can be consumed collectively across your global account. This is typical of global accounts that have the **subscription-based** commercial model agreement.
- **Unlimited quota:** There is no upper limit to how much the service plan can be used collectively across your global account. This is typical of most services that are eligible to global accounts with a **consumption-based** commercial model agreement (Cloud Platform Enterprise Agreement (CPEA) and Pay-As-You-Go for SAP BTP). At the end of the month, the number of units used determines how much you are charged.

Regardless of whether the global quota of a plan on the global account level is fixed or unlimited, the specifics and properties of each plan determines how its quota can be distributed by the global account admin to the subaccounts in your global account. There are two types of quota assignments:

- **Numeric assignments:** The global account admin assigns the plan to a subaccount and specifies the maximum quantity of units from the global quota that can be consumed by the subaccount. Admins can use this assignment to limit users from exceeding a specific quota for cost-controlling purposes. Typically, but not always, the assigned quota is directly related to the number of instances that can be created in subaccounts for the assigned plan.
- **Non-numeric assignments:** The global account admin does not assign a specific quantity but simply grants access to the specific plan by assigning the plan to the subaccount. When the plan is assigned, it's available for use by the subaccount. SaaS application assignments are typically non-numeric, whereby a subaccount can either subscribe to the app or not. For services that fall into this category of non-numeric assignments, there is usually no limit to the number of instances that can be created.

See also the sections below and [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

## Distribution and Usage of Entitlements and Quotas

Entitlements and quotas are purchased and managed at global account level, from where they're distributed to subaccounts, which consume them.

### → Tip

You can also distribute entitlements to directories, and then redistribute to the subaccounts under the directories; however, this is optional. By default, directories are not enabled to manage entitlements. This feature needs to be enabled per directory by the global account administrator.

When assigning entitlements and quotas to directories, you also have the option to automatically assign a set amount of quota to each new subaccount added to the directory. This option doesn't apply to subaccounts that are already in the directory when you select this option. The quota you assigned to the directory is then gradually distributed to all subaccounts that you add to that directory, until it runs out. Once the directory quota runs out, if you add a new subaccount to that directory it won't get any quota automatically anymore.

Since directories are only a way of grouping subaccounts, you can't consume a service at directory level. However, when you assign entitlements and quotas from the global account to a directory, the quota you assigned is shown as used, even if there are no subaccounts in that directory to consume the quota. You can think of it as a way to "reserve" quota and make sure it's not assigned to other subaccounts or directories.

When you remove quotas or assignments from a directory or subaccount, they become available again at global account level and can be assigned to other directories or subaccounts; unless the quota is reserved for a given directory then the freed quota remains available only to that directory and its subaccounts.

### ⓘ Note

Before a subaccount admin can enable a quota-based environment, such as Kyma, the subaccount admin must first assign the environment as an entitlement to the subaccount. Other environments, such as Cloud Foundry, are available by default to all subaccounts, and therefore are not available as entitlements.

For more information, see:

- [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)
- [Configure Entitlements and Quotas for Directories \[page 2188\]](#)

## Quota Plans

In the Cloud Foundry environment, you can further distribute the quotas that are allocated to a subaccount. This is done by creating space quota plans and assigning them to the spaces.

Space quota plans are optional and are used to limit how much each space can use, not to enable. If you don't create any space quota plans, all spaces in a subaccount using the Cloud Foundry environment have access to all entitlements and quotas in that subaccount. This means that one space could use up all the quota in a subaccount. If you want to prevent that from happening and set a limit to how much each space can use from the total quota available in the subaccount, you can use quota plans and assign them to spaces.

For more information on space quota plans in the Cloud Foundry environment, see:

- <https://docs.cloudfoundry.org/adminguide/quota-plans.html> ↗
- [Managing Space Quotas \[page 2447\]](#)

## Resource Providers

SAP BTP allows you to connect your global account in the SAP BTP Cockpit to your provider account from a non-SAP cloud vendor, and consume remote service resources that you already own and which are supported by SAP through this channel.

For example, if you're subscribed to Amazon Web Services (AWS) and have already purchased services, such as PostgreSQL, you can register the vendor as a resource provider in SAP BTP and consume this service across your subaccounts together with other services offered by SAP.

SAP BTP currently supports the following vendors and their consumable services:

Cloud Vendor	Supported Services
Amazon Web Services	<a href="#">Amazon Relational Database Service (RDS) - PostgreSQL</a>
Microsoft Azure	<a href="#">Azure Database for PostgreSQL</a> .

After you configure a new resource provider, its supported services are added as entitlements in your global account. In the  [Entitlements](#) page in the cockpit, you can then allocate the required services and quotas to the relevant directories and subaccounts in your global account.

For more information, see [Managing Resource Providers \[page 2147\]](#).

## Related Information

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

**Tutorial:** [Manage Entitlements on SAP BTP Trial](#) ↗

**Troubleshooting:** [Entitlement and Quota Problems](#) ↗

## 2.8 User and Member Management

On SAP BTP, user management takes place at all levels from global account to environment. There are different types of users, such as depending on their roles in the company.

### User Accounts

A user account corresponds to a particular user in an identity provider. The user is always stored in an external identity provider, such as a custom tenant of SAP Cloud Identity Services - Identity Authentication or the default identity provider.

**User accounts** enable users to log on to SAP BTP, access subaccounts, and to use applications according to the permissions granted to them.

#### Note

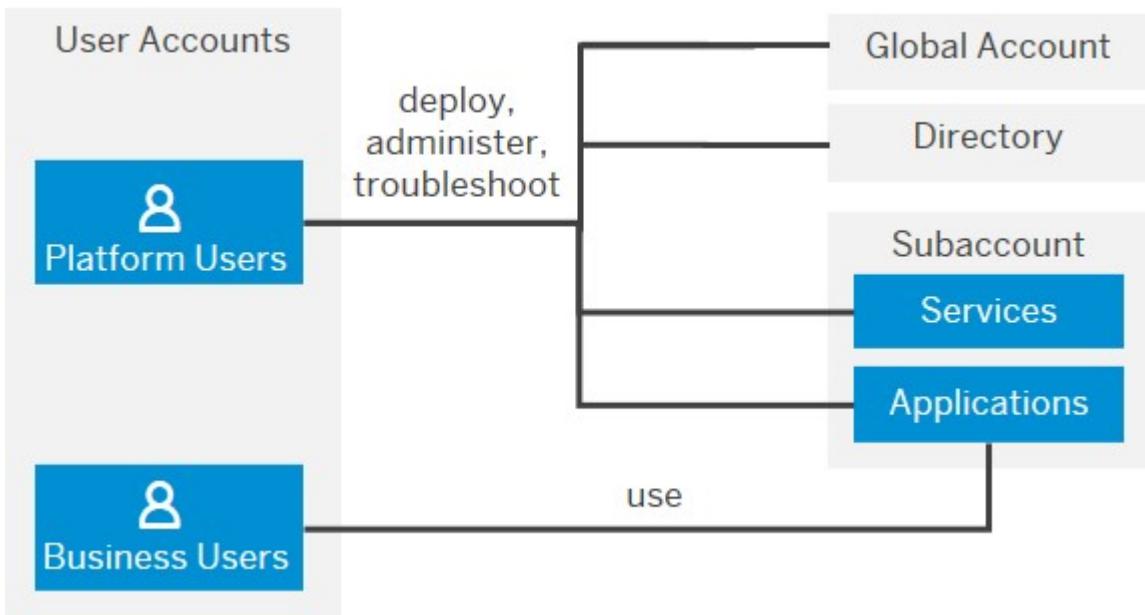
A user name alone doesn't determine a concrete user account with associated authorizations, as you can have users with the same user name in different identity providers. Accessible data and allowed operations also depend on the identity provider. The concrete user is identified by the combination of user name and identity provider.

#### Example

There are two users with the same user name, which is the email address here. The two users with different identity providers have different authorizations and can access different applications.

- julia.moore@acme.com from the custom identity provider has authorizations to access her favorite industrial applications. She needs the logon with the custom identity provider for her actual work.
- julia.moore@acme.com from the default identity provider has no authorizations.

Before diving into the different user and member management concepts, it's important to understand the difference between the two types of users we're referring to: **platform users** and **business users**.



For more information, see [Platform Users \[page 105\]](#) and [Business Users \[page 119\]](#).

## Related Information

[Working with Users \[page 2265\]](#)

[Roles and Role Collections \[page 2285\]](#)

[Attributes \[page 2288\]](#)

[Trust and Federation with Identity Providers \[page 2204\]](#)

## 2.8.1 Platform Users

**Platform users** are usually developers, administrators or operators who deploy, administer, and troubleshoot accounts, applications and services on SAP BTP. They're the users that have full access and give certain permissions, for instance, at global account, directory, or subaccount level. Members only have basic access.

Platform users who have administrative permissions can view or manage the list of global accounts, subaccounts, and environments, such as Cloud Foundry orgs and spaces. Members have basic access to them using the SAP BTP cockpit, the SAP BTP command-line interface (btp CLI), or environment-specific CLI, such as the Cloud Foundry (CF) CLI.

For platform users, there's a [default identity provider \[page 2258\]](#). We expect that you have your own identity provider. We recommend that you configure your custom tenant of SAP Cloud Identity Services as the identity provider and connect SAP Cloud Identity Services to your own corporate identity provider.

### ⓘ Note

For China (Shanghai) and Government Cloud (US) regions, a different default identity provider is used, and you can't use SAP Cloud Identity Services as identity provider in the global account.

If you want to use two-factor authentication in the China (Shanghai) region, see this [blog article](#) on SAP Community.

## Member Management

**Member management** refers to managing permissions for platform users. Members have only basic access to SAP BTP.

Member management happens at global account, directory, subaccount, and environment level. Members' permissions apply to all operations that are associated with the global account, the organization, or the space, irrespective of the tool used. Depending on the scope and the cloud management tools you're using, you manage members in different ways:

Global Accounts	Directories	Subaccounts
<p>You manage global account members by assigning role collections to platform users. Use the following predefined role collections:</p> <ul style="list-style-type: none"><li>• Global Account Administrator</li><li>• Global Account Viewer</li></ul> <p>Assign these role collections from the SAP BTP cockpit or the btp CLI.</p> <p>See:</p> <p><a href="#">Role Collections and Roles in Global Accounts, Directories, and Subaccounts [page 107]</a></p> <p><a href="#">Add Members to Your Global Account [page 2146]</a></p> <p><a href="#">Create Users [page 2270]</a></p>	<p>You manage directory members by assigning role collections to platform users. Use the following predefined role collections:</p> <ul style="list-style-type: none"><li>• Directory Administrator</li><li>• Directory Viewer</li></ul> <p>Assign these role collections from the SAP BTP cockpit or the btp CLI.</p> <p>See:</p> <p><a href="#">Role Collections and Roles in Global Accounts, Directories, and Subaccounts [page 107]</a></p> <p><a href="#">Create Users [page 2270]</a></p> <p><a href="#">Manage Users in Directories [page 2169]</a></p>	<p>You manage subaccount members by assigning role collections to platform users.</p> <p><b> ⓘ Note</b></p> <p>Neo subaccounts don't use role collections.</p> <p>For more information, see <a href="#">Managing Member Authorizations in the Neo Environment</a>.</p> <p>Use the predefined role collections, such as:</p> <ul style="list-style-type: none"><li>• Subaccount Administrator</li><li>• Subaccount Viewer</li></ul> <p>Assign these role collections from the SAP BTP cockpit or the btp CLI.</p> <p>See:</p> <p><a href="#">Role Collections and Roles in Global Accounts, Directories, and Subaccounts [page 107]</a></p> <p><a href="#">Add Members to Your Subaccount [page 2177]</a></p> <p><a href="#">Create Users [page 2270]</a></p>

## Member Management in the Cloud Foundry Environment

Orgs	Spaces
<p>Manage org members on the <a href="#">Members</a> page at environment level in the SAP BTP cockpit or with the Cloud Foundry CLI.</p> <p>A platform user added as an org member can be either an <b>Org Manager</b> or an <b>Org Auditor</b> or implicitly an <b>Org User</b>.</p> <p>See:</p> <p><a href="#">About Roles in the Cloud Foundry Environment [page 2431]</a></p> <p><a href="https://docs.cloudfoundry.org/concepts/roles.html#roles">https://docs.cloudfoundry.org/concepts/roles.html#roles</a></p> <p><a href="#">Add Org Members [page 2437]</a></p>	<p>Manage space members on the <a href="#">Members</a> page at space level in the SAP BTP cockpit or with the Cloud Foundry CLI.</p> <p>A platform user added as a space member can be either a <b>Space Manager</b>, <b>Space Developer</b>, <b>Space Auditor</b>, or <b>Space Supporter</b>.</p> <p>See:</p> <p><a href="#">About Roles in the Cloud Foundry Environment [page 2431]</a></p> <p><a href="https://docs.cloudfoundry.org/concepts/roles.html#roles">https://docs.cloudfoundry.org/concepts/roles.html#roles</a></p> <p><a href="#">Add Space Members [page 2444]</a></p>

For more information on the Kyma environment, see [Assign Roles in the Kyma Environment \[page 3137\]](#).

See also [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

### 2.8.1.1 Role Collections and Roles in Global Accounts, Directories, and Subaccounts

SAP BTP provides a set of role collections to set up administrator access to your global account and subaccounts.

Role collections group authorizations for resources and services. Your administrators assign these role collections to other platform users to create new administrators. Role collections consist of individual roles. For more information on role collections, roles, see the related link.

Role collections are account-specific. Role collections that exist in the global account don't exist in the subaccounts. Likewise, role collections in subaccounts aren't available in the global account.

#### ⓘ Note

Neo subaccounts don't use role collections.

For more information, see [Managing Member Authorizations in the Neo Environment](#).

## Role Collections

You can use the default role collections, but you can't change or delete them. SAP BTP provides the following administrator role collections:

- *Global Account Administrator*
- *Subaccount Administrator*

- *Directory Administrator*
- *Cloud Connector Administrator*
- *Connectivity and Destination Administrator*
- *Destination Administrator*
- *Subaccount Service Administrator*

SAP BTP provides also viewer role collections for the global account and for subaccounts. In contrast to the administrator role collections, viewer role collections only grant read access.

- *Global Account Viewer*
- *Subaccount Viewer*
- *Directory Viewer*

## Administrator Role Collections

If you assign the *Global Account Administrator* role collection to a user, this user can perform administration tasks for subaccounts, role collections, identity providers, entitlements, and regions on the level of the global account. If you assign the *Global Account Viewer* role collection, this user can view subaccounts, role collections, identity providers, entitlements, and regions on the level of the global account.

### Note

You can also use the command-line interface (CLI) for SAP BTP to assign authorizations. For more information on managing authorizations, see the related link.

### Global Account Administrator Role Collection

Roles Included	Description
Global Account Admin	<p>Role for global account members with read-write authorizations for core commercialization operations, such as updating global accounts, setting entitlements, and creating, updating, and deleting subaccounts.</p> <p>The <i>GlobalAccount_Admin</i> role template contains this role. You find the role template in the SAP BTP Cockpit if you choose the <i>cis-central!&lt;suffix&gt;</i> application identifier.</p>
Global Account Usage Reporting Viewer	<p>Role for global account members with read-only authorizations for core commercialization operations, such as viewing global account usage information.</p> <p>The <i>GlobalAccount_Usage_Reportin</i> role template provides this role. You find the role template in the SAP BTP Cockpit if you chose the <i>uas!&lt;suffix&gt;</i> application identifier</p>

Roles Included	Description
User and Role Administrator	<p>Manage authorizations, trusted identity providers, and users.</p> <p>The <code>xsuaa_admin</code> role template provides this role. You find the role template in the SAP BTP Cockpit if you choose the <code>xsuaa!&lt;suffix&gt;</code> application identifier.</p> <div data-bbox="811 579 1391 682" style="background-color: #f0f0f0; padding: 10px;"> <span data-bbox="811 579 865 617">❖ Example</span>  <code>xsuaa!tl</code> </div>
System Landscape Administrator	<p>Administrative access to systems and scenario-related resources.</p> <p>The <code>GlobalAccount_System_Landscape_Administrator</code> role template provides this role. You find the role template in the SAP BTP Cockpit if you choose the <code>cmp!&lt;suffix&gt;</code> application identifier.</p> <div data-bbox="811 952 1391 1055" style="background-color: #f0f0f0; padding: 10px;"> <span data-bbox="811 952 865 990">❖ Example</span>  <code>cmp!b7</code> </div>

If you assign the `Subaccount Administrator` role collection to a user, you grant a user administration permission for a subaccount.

#### Subaccount Administrator Role Collection

Roles Included	Description
Cloud Connector Administrator	Operate the data transmission tunnels used by the Cloud connector.
Destination Administrator	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Subaccount Admin	Role for subaccount members with read-write authorizations for core commercialization operations, such as viewing subaccount entitlements, and creating and deleting environment instances.
Subaccount Service Administrator	Administrative access to service brokers and environments on a subaccount level.
User and Role Administrator	Manage authorizations, trusted identity providers, and users.

If you assign the `Cloud Connector Administrator` role collection to a user, you grant the user administration permissions for the Cloud Connector in a subaccount.

## Cloud Connector Administrator Role Collection

Roles Included	Description
Cloud Connector Administrator	Operate the data transmission tunnels used by the Cloud connector.

If you assign the *Connectivity and Destination Administrator* role collection to a user, you grant the user administration permissions for the Cloud Connector and SAP Destination service in a subaccount.

## Connectivity and Destination Administrator Role Collection

Roles Included	Description
Cloud Connector Administrator	Operate the data transmission tunnels used by the Cloud connector.
Destination Administrator	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.

If you assign the *Destination Administrator* role collection to a user, you grant the user administration permissions for the SAP Destination service in a subaccount.

## Destination Administrator Role Collection

Roles Included	Description
Destination Administrator	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.

If you assign the *Subaccount Service Administrator* role collection to a user, you grant the user administration permissions for the Service Manager in a subaccount.

## Subaccount Service Administrator Role Collection

Roles Included	Description
Subaccount Service Administrator	Administrative access to service brokers and environments on a subaccount level.

## Viewer Role Collections

If you assign the *Global Account Viewer* role collection to a user, you grant read access to the same information as the Global Account Administrator role collection.

## Global Account Viewer Role Collection

Roles Included	Description
Global Account Viewer	Role for global account members with read-only authorizations for core commercialization operations, such as viewing global accounts, subaccounts, entitlements, and regions.
Global Account Usage Reporting Viewer	Role for global account members with read-only authorizations for core commercialization operations, such as viewing global account usage information.
User and Role Auditor	Read-only access for authorizations, trusted identity providers, and users.
System Landscape Viewer	Viewer access to systems and scenario-related resources.

If you assign the *Subaccount Viewer* role collection to a user, you restrict a user's viewer permission to the subaccounts.

## Subaccount Viewer Role Collection

Roles Included	Description
Cloud Connector Auditor	View the data transmission tunnels used by the Cloud connector to communicate with back-end systems.
Destination Viewer	View destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Subaccount Service Auditor	Read-only access to service brokers and environments on a subaccount level
Subaccount Viewer	Role for subaccount members with read-only authorizations for core commercialization operations, such as viewing subaccount entitlements, details of environment instances, and job results.
User and Role Auditor	Read-only access for authorizations, trusted identity providers, and users.

### ⓘ Note

To obtain permissions to view the service binding credentials, ask an administrator to create a new role collection that includes the Subaccount Viewer role collection and the Service Credentials Viewer role (see [Define a Role Collection \[page 2275\]](#)).

## Directory Role Collections

The role collections *Directory Administrator* and *Directory Viewer* can be assigned after the creation of a directory. If you select the *Enable user management* checkbox, you receive the *Directory Administrator* role collection by default. You can't create custom role collections for directories.

The *Directory Administrator* role collection grants a user administration permission for directories.

Directory Administrator Role Collection

Roles Included	Description
Directory Admin	Role for directory members with read-write authorizations for core commercialization operations, such as updating directories, setting entitlements, and creating, updating, and deleting subaccounts.
Directory Usage Reporting Viewer	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directory usage information.
User and Role Administrator	Manage authorizations, trusted identity providers, and users.

The *Directory Viewer* role collection grants a user read access to the same information as the Directory Administrator role collection.

Directory Viewer Role Collection

Roles Included	Description
Directory Usage Reporting Viewer	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directory usage information.
Directory Viewer	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directories, subaccounts, entitlements, and regions.
User and Role Auditor	Read-only access for authorizations, trusted identity providers, and users.

## Roles

Create your own role collections by using the roles of the default role collections. The roles are based on role templates, which are provided by applications. The application identifier refers to the application, which provides the role templates.

### Example

The *GlobalAccount\_Admin* role template provides the *Global Account Admin* role. You find the role template in the cockpit if you choose the *cis-central!<suffix>* application identifier.

## Example

cis-central!b1

The following table provides the information about the roles that are available.

### Role Details

Roles	Available in	Role Templates	Application Identifier
Cloud Connector Administrator	Subaccount	Cloud_Connector_Administrator	connectivity!<suffix>
Cloud Connector Auditor	Subaccount	Cloud_Connector_Auditor	connectivity!<suffix>
Destination Administrator	Subaccount	Destination_Administrator	destination-xsappname!<suffix>
Destination Viewer	Subaccount	Destination_Visitor	destination-xsappname!<suffix>
Directory Admin	Directory	Directory_Visitor	cis-central!<suffix>
Directory Usage Reporting Viewer	Directory	Directory_Usage_Report- ing_Visitor	uas!<suffix>
Directory Viewer	Directory	Directory_Admin	cis-central!<suffix>
Global Account Admin	Global account	GlobalAccount_Admin	cis-central!<suffix>
Global Account Viewer	Global account	GlobalAccount_Visitor	cis-central!<suffix>
Global Account Usage Re- porting Viewer	Global account	GlobalAccount_Usage_Re- porting_Visitor	uas!<suffix>
Service Credentials Viewer	Subaccount	Service_Credentials_Visitor	service-manager!<suffix>
System Landscape Adminis- trator	Global account	GlobalAccount_Sys- tem_Landscape_Administra- tor	cmp!<suffix>
System Landscape Viewer	Global account	GlobalAccount_Sys- tem_Landscape_Visitor	extension-service-cmp!<suffix>
Subaccount Admin	Subaccount	Subaccount_Admin	cis-local!<suffix>
Subaccount Viewer	Subaccount	Subaccount_Visitor	cis-local!<suffix>
Subaccount Service Admin- istrator	Subaccount	Subaccount_Service_Admin- istrator	service-manager!<suffix>
Subaccount Service Auditor	Subaccount	Subaccount_Service_Auditor	service-manager!<suffix>

Roles	Available in	Role Templates	Application Identifier
User and Role Administrator	Global account and subaccount	xsuaa_admin	xsuaa!<suffix>
User and Role Auditor	Global account and subaccount	xsuaa_auditor	xsuaa!<suffix>

## Roles in Environments

If you've enabled environments in your subaccount, manage the roles for those environments. For more information, see:

- [About Roles in the Cloud Foundry Environment \[page 2431\]](#)
- [Assign Roles in the Kyma Environment \[page 3137\]](#)

### ⓘ Note

The ABAP environment uses the roles of the SAP BTP, Cloud Foundry Environment.

## Related Information

- [User and Member Management \[page 104\]](#)
- [Working with Role Collections \[page 2274\]](#)
- [Building Roles and Role Collections for Applications \[page 2284\]](#)
- [Mapping Role Collections in the Subaccount \[page 2281\]](#)
- [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)

## 2.8.1.2 Bringing Your Corporate Identity Provider for Platform Users

SAP BTP supports the use of your own identity provider for platform users.

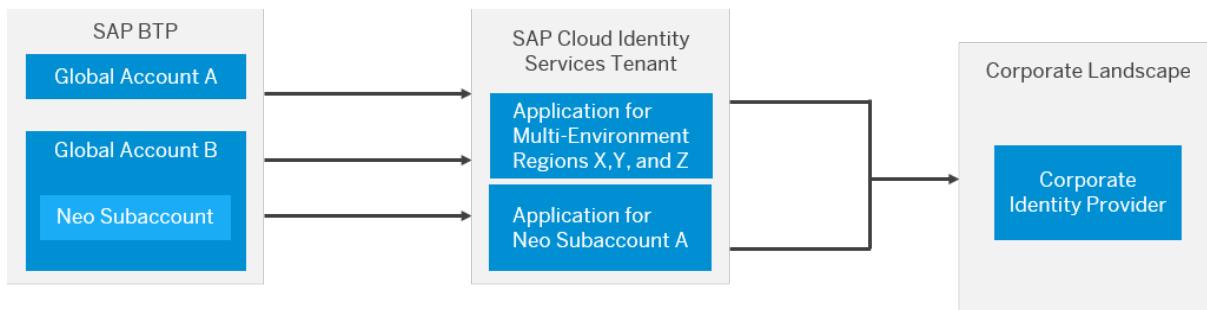
Platform users perform technical development, operations, and administration tasks. They manage global accounts, directories, and subaccounts using the SAP BTP cockpit and the SAP BTP CLI. They also develop and operate custom applications. By hosting these users in your own identity provider, you gain a number of advantages over hosting them in SAP ID service or in SAP Universal ID.

- Integrate the management of these users with your broader identity management strategy, hosted on your own identity providers. You control your own user lifecycle and single sign-on strategies throughout the entire landscape.

- Enforce your own password and authentication policies, such as stronger passwords or multifactor authentication.

The following figure illustrates the architecture required for platform users. This configuration is independent of the default configuration with SAP ID service. You can continue to use SAP ID service in parallel.

**Authentication Architecture for Platform Users with a Corporate Identity Provider**



In the preceding figure, you enable trust between the SAP BTP global account and your corporate identity provider over your tenant of SAP Cloud Identity Services - Identity Authentication. Trust is configured differently in the Kyma environment (see the related link). For each global account, you choose the SAP Cloud Identity Services tenant to use as the identity provider for platform users. For the platform identify provider, you can have up to three SAP Cloud Identity Services tenants per global account. Multiple global accounts can share the same SAP Cloud Identity Services tenant. When you log on to a platform resource, such as the cockpit, you indicate the SAP Cloud Identity Services tenant that you want to log on with. For example, to log on to the cockpit, copy a URL parameter from the cockpit to identify the tenant:

`https://cockpit.btp.cloud.sap/cockpit/?idp=<tenant>.accounts.ondemand.com`

For example: `https://cockpit.btp.cloud.sap/cockpit/?idp=cidppuxhm.accounts.ondemand.com`

Once you've logged on, the cockpit displays any global accounts and subaccounts of which your platform user is a member.

A user identifier alone isn't enough to identify a user. Additionally, you also need an identifier for an identity provider. The reason is that the system treats users with the same name but from different identity providers as separate users.

For global accounts, directories, multi-environment subaccounts, and the Cloud Foundry environment, the user identifier is always an email address, and the identifier of the identity provider is the origin key (alias for the trust configuration). For example, a platform user in the default identity provider, SAP ID service, and another user in your corporate identity provider with the same email address might have different authorizations.

### Example

There are two users with the same email address. It is common practise that the two users with different identity providers have different authorizations and can access different applications.

- `julia.moore@acme.com` with the default identity provider isn't authorized for any subaccount.
- `julia.moore@acme.com` with the `acme-platform` custom identity provider is the administrator for multiple subaccounts. The subaccounts are only visible when she logs on with the custom identity provider.

You can log on to the cockpit with both, but the cockpit displays different user information. This difference is because you've logged on with different identity providers.

For Neo subaccounts, it is, by default, the P user ID for local SAP Cloud Identity Services users or whatever the corporate identity provider connected to the SAP Cloud Identity Services tenant sends as subject. The identifier of the identity provider is the default domain of the SAP Cloud Identity Services tenant, for example acme.accounts.ondemand.com. It is called "user base". For more information, see [Platform Identity Provider](#).

You also see this difference when assigning roles. You must provide the origin or user base in addition to the email address or user ID of the user. When platform users use the Cloud Foundry command-line interface or service dashboards, they need to remember the origin. You can choose your own origin, but the origin must be unique across all customers. We recommend that you use a meaningful name that helps identify the target it points to.

In SAP Cloud Identity Services - Identity Authentication in Neo subaccounts, there's one application that represents SAP BTP overall. So, if you have multiple global accounts with the same SAP Cloud Identity Services tenant, they all share the same application in your SAP Cloud Identity Services tenant. Here customers typically configure settings such as the corporate identity provider used for authentication and user attribute mapping between systems. For more information, see [Map User Attributes from a Corporate Identity Provider for Platform Users \[page 2244\]](#).

## Related Information

[Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface \[page 2251\]](#)

[Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#)

[Configure a Custom Identity Provider for Kyma \[page 3134\]](#)

### 2.8.1.3 Impact of Upgrading from Feature Set A to Feature Set B on User and Account Management

While we've tried to make upgrading your account management feature set as simple as possible, we've had to make a few adjustments. To help you find your way, we've gathered a summary of changes you can expect in your accounts.

#### ⓘ Note

For more information about the upgrade process itself, see SAP Note [3027721](#).

## Reactivation of the Default Identity Provider for Applications

If you deactivated the default identity provider, SAP ID service, for business users in multi-environment subaccounts in cloud management tools feature set A, the process of upgrading to cloud management tools

feature set B reactivates the default identity provider. We reactivate the default identity provider because cloud management tools feature set B also uses it for platform users.

#### → Recommendation

Keep at least one user from the default identity provider in each account to still have access if there are ever issues with a custom identity provider.

Even though the default identity provider has been reactivated, the option to authenticate with the default identity provider is hidden from business users. Business users are redirected to your custom identity provider. Hiding the default identity provider ensures that the user experience for your business users remains the same. Users from the default identity provider can log on, only if the users are authorized, in other words, users with existing shadow users.

For more information about shadow users, see [Working with Users \[page 2265\]](#).

#### ⓘ Note

Whether the default identity provider was reactivated or not, applications that share the default identity provider, such as SAP Support Portal or your demo application in your subaccount, no longer require reauthentication, when you switch from one application to the other. This change improves the single sign-on experience.

## Custom Identity Providers for Platform Users

In cloud management tools feature set B, you're free to integrate custom identity providers for platform users. If you use custom identity providers for platform users in cloud management tools feature set A, see SAP Note [3264627](#).

## Global Account Users Identified by E-Mail Address

With cloud management tools feature set B, global account users from the SAP ID service are identified by their e-mail address and not their user ID. If you've multiple user accounts that share the same e-mail address, they all get the same authorizations.

## Mapping Account Authorizations Between Feature Set A and Feature Set B

The following table lists the role collections for account administration that a user has in cloud management tools feature set B, based on the role memberships the user had in cloud management tools feature set A.

## Account Authorization Mappings Between Feature Set A and Feature Set B

Authorizations in Feature Set A	Authorizations in Feature Set B	More Information
Administrator of global account	<i>Global Account Administrator</i> in global account and <i>Subaccount Administrator</i> in multi-environment subaccounts	In cloud management tools feature set A, members of the global account have global account administrator privileges. Such users can create and manage subscriptions for subaccounts.
Security administrator in multi-environment subaccount	<i>Subaccount Administrator</i> in multi-environment subaccount	<p>→ Tip</p> <p>To ensure that your global account administrators still have these authorizations, these users are also assigned subaccount administrator authorizations in <b>all</b> multi-environment subaccounts of the global account. After the move, consider if your global account administrators really need subaccount administrator access to all your subaccounts.</p>

## Mapping Cloud Foundry Authorizations Between Feature Set A and Feature Set B

The following table lists the roles and role collections a user receives, based on the Cloud Foundry roles the user had in cloud management tools feature set A.

Mappings of Cloud Foundry Authorizations Between Feature Set A and Feature Set B

Authorizations in Feature Set A	Authorizations in Feature Set B	More Information
Org Manager	Org Manager	In cloud management tools feature set A, a user with the <i>Org Manager</i> , <i>Space Manager</i> , or <i>Space Developer</i> roles could also manage cloud connectors and destinations. To make sure that such users don't lose any authorizations, we check the users authorizations and add the required role collections.
Space Manager	Space Manager	
Space Developer	Space Developer	
	<b> ⓘ Note</b> If the user doesn't already have the <i>Subaccount Administrator</i> role collection, the user receives the <i>Connectivity and Destination Administrator</i> role collection.	
Org Auditor	Org Auditor	No change.
Space Auditor	Space Auditor	

## Role Collection Names

Any role collection that you created in cloud management tools feature set A with a name that is a reserved role collection name in cloud management tools feature set B, will have the suffix "(Custom)" appended to your role collection's name. For example, *Subaccount Administrator* will be renamed to *Subaccount Administrator (Custom)* after the upgrade.

## No Impact on Neo Subaccounts

The authorization model of Neo subaccounts remains the same in both feature sets.

### 2.8.2 Business Users

**Business users** use the applications that are deployed to SAP BTP. For example, the end users of SaaS apps or services, such as SAP Build Work Zone, or end users of your custom applications are business users.

Application developers (platform users) create and deploy application-specific security artifacts for business users, such as scopes. Administrators use these artifacts to assign roles, build role collections, and assign these role collections to business users or user groups. In this way, they control the users' permissions in the application.

For business users, there's a [default identity provider \[page 2258\]](#), too. We expect that you have your own identity provider. We recommend that you configure your custom tenant of SAP Cloud Identity Services as the identity provider and connect SAP Cloud Identity Services to your own corporate identity provider.

## User Management

**User management** refers to managing authentication and authorization for your business users.

To manage your business users:

1. Configure trust to an application identity provider in your subaccount.
2. Create shadow users in your subaccount for your business users in your identity provider.  
When a user accesses a resource, SAP BTP redirects the user to the identity provider for authentication.  
You assign authorizations to shadow users in SAP BTP.
3. Assign role collections either directly to users or map them to user groups.  
The role collections were either delivered from the applications to which you subscribed or custom developed by your team.

To learn more about user management, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#).

## 2.9 Tools, Programming Models, Programming Languages, and APIs

SAP BTP provides various programming languages and tools for your development project.

### [Tools \[page 121\]](#)

SAP BTP includes many tools to help you develop and manage applications, and connect them to your on-premise systems.

### [Programming Languages \[page 123\]](#)

SAP BTP supports many different programming languages; the availability of each depends on the development environment you're using.

### [Programming Models \[page 124\]](#)

### [Continuous Integration and Delivery \(CI/CD\) \[page 125\]](#)

Configure and run predefined continuous integration and delivery (CI/CD) pipelines that automatically build, test, and deploy your code changes to speed up your development and delivery cycles.

### [APIs \[page 128\]](#)

Discover and consume APIs to manage, build, and extend the core capabilities of SAP BTP.

### [Cloud Management Tools – Feature Set Overview \[page 129\]](#)

Cloud management tools represent the group of technologies designed for managing SAP BTP.

### [Prerequisites and Restrictions \[page 141\]](#)

Find a list of the product prerequisites and restrictions for SAP BTP.

## 2.9.1 Tools

SAP BTP includes many tools to help you develop and manage applications, and connect them to your on-premise systems.

The availability of tools can depend on the cloud management tools feature set that you are running on. For more information, see [Cloud Management Tools — Feature Set Overview \[page 129\]](#).

Tool	Description
<a href="#">Account Administration in the Cockpit [page 2140]</a>	<p>The SAP BTP cockpit is the web-based administration interface of SAP BTP and provides access to a number of functions for configuring and managing applications, services, and subaccounts. Use the cockpit to manage resources, services, security, monitor application metrics, and perform actions on cloud applications.</p> <p>Used for account administration in SAP BTP.</p>
<a href="#">ABAP Development Tools for Eclipse</a>	<p>The ABAP Development Tools for Eclipse enables you to perform ABAP-based development tasks, when you want to build, to extend, and to run ABAP Cloud applications. It supports features like syntax check, code completion, syntax highlighting, pretty printing, navigation, search, quick fixes.</p> <p>Used for programming in ABAP.</p>
<a href="#">SAP Business Application Studio</a>	<p>Designed and optimized for business application development in SAP ecosystems, SAP Business Application Studio enhances productivity by offering specialized tools for various scenarios, including SAP Fiori application development, SAP HANA native extensions, full-stack and mobile application development, and more.</p> <p>Central to the development environment is Code-OSS, the open-source foundation of Visual Studio Code, ensuring a familiar experience for developers when creating SAP-centric applications. SAP Business Application Studio streamlines the building, testing, and deployment of applications with integrated features for source control and testing. Furthermore, its Full-Stack Application Productivity Toolkit offers intuitive visual tools covering the entire development process, guaranteeing seamless integration with various SAP services and solutions.</p> <p>Used for:</p> <ul style="list-style-type: none"><li>• SAP Fiori development for ABAP and non-ABAP</li><li>• CAP full-stack application development</li></ul>

Tool	Description
SAP Build	<p>SAP Build enables everyone, no matter the skill level, to create and augment enterprise applications, process automations, and business sites with drag-and-drop simplicity.</p> <p>Used for low-code, no-code development.</p>
Command Line Interface for Cloud Foundry [page 2452]	<p>The Cloud Foundry command line interface enables you to work with the Cloud Foundry runtime to deploy and manage your applications.</p> <p>Used for managing subaccounts in the SAP BTP, Cloud Foundry runtime.</p>
SAP BTP Command Line Interface [page 2321]	<p>The SAP BTP command line interface (btp CLI) is the command line tool for convenient account management, such as managing global accounts, directories, subaccounts, entitlements, environment instances, multitenant application subscriptions, and users and their authorizations.</p> <p>Used for managing global accounts in SAP BTP.</p>
Cloud Connector	<p>The Cloud Connector serves as the link between on-demand applications in SAP BTP. This is the browser-based and existing on-premise systems. You can control the resources available for the cloud applications in those systems.</p> <p>Used for connectivity between SAP BTP applications and on-premise systems.</p>
Terraform provider for SAP BTP	<p>The Terraform provider for SAP BTP enables you to automate the provisioning, management, and configuration of resources on SAP BTP.</p>
kubectl	<p>The Kubernetes command line tool to communicate with a Kubernetes cluster's control plane, using the Kubernetes API.</p> <p>Used as a Kubernetes tool connected with the SAP BTP, Kyma runtime.</p>
kubelogin	<p>A kubectl plugin for Kubernetes OpenID Connect (OIDC) authentication.</p> <p>Used as a Kubernetes tool connected with the SAP BTP, Kyma runtime.</p>
Helm	<p>The package manager for Kubernetes, used for installing and managing Kubernetes applications in form of Helm charts.</p> <p>Used as a Kubernetes tool connected with the SAP BTP, Kyma runtime.</p>

Tool	Description
Paketo (Pack) 	<p>Pack is a tool maintained by the Cloud Native Buildpacks project to support the use of buildpacks. Pack lets you build container images, which are collaboratively maintained making it easier to maintain and update.</p> <p>Used as a Kubernetes tool connected with the SAP BTP, Kyma runtime.</p>
Docker Desktop 	<p>Docker Desktop is an application that enables you to manage (build, push, pull, and run) container images on your desktop and a docker-compatible command line interface.</p> <p>Used as a Kubernetes tool connected with the SAP BTP, Kyma runtime.</p>
SAP Cloud SDK	<p>SAP Cloud SDK provides a layer of abstractions for features of SAP BTP such as logging, multitenancy, and connectivity. It also includes project templates for different execution environments and implementations.</p> <p>Used for programming in Java and JavaScript.</p>
Eclipse Tool for the Cloud Foundry Runtime 	<p>The Eclipse plug-in for the Cloud Foundry runtime is a Java-based toolkit for Eclipse IDE that enables you to develop and deploy Java and Spring applications in the Cloud Foundry runtime from Eclipse or Spring Tool Suite, as well as perform operations such as logging, managing user roles, creating connectivity destinations, and so on.</p> <p>Used for programming in Java in the SAP BTP, Cloud Foundry runtime.</p>
Service-Specific Tools	<p>The services that run on SAP BTP can come with service-specific tools. For an overview of the services and their tools, see the <a href="#">SAP Discovery Center</a> .</p>

## 2.9.2 Programming Languages

SAP BTP supports many different programming languages; the availability of each depends on the development environment you're using.

The following programming languages are available in the environments:

Supported Environments and Programming Languages

Environment	Programming Language
Cloud Foundry environment	<a href="#">Java [page 261]</a>

Environment	Programming Language
	<a href="#">Node.js [page 343]</a>
	<a href="#">Python [page 354]</a>
ABAP environment	<a href="#">ABAP [page 705]</a>
Kyma environment	<a href="#">Serverless Functions: Node.js and Python [page 1977]</a>
	Microservices: Any cross-platform programming language

## 2.9.3 Programming Models

### ABAP Cloud

ABAP Cloud reflects the modern way to develop ABAP. It allows you to build lifecycle-stable and cloud-ready business applications, services, and extensions.

ABAP Cloud provides tools and techniques that ensure cloud qualities, promotes new technologies, contains a cloud-optimized subset of the ABAP language, and makes upgrade cycles easier by a clear separation between custom code and SAP code by only using released APIs and objects. See [ABAP Cloud](#) in [SAP BTP Developer's Guide](#).

### SAP Cloud Application Programming Model (CAP)

The SAP Cloud Application Programming Model (CAP) is a framework of languages, libraries, and tools for building enterprise-grade services and applications. It supports Java (with Spring Boot), JavaScript, and TypeScript (with Node.js), which are some of the most widely adopted languages. CAP guides developers along a path of proven best practices and a great wealth of out-of-the-box solutions to recurring tasks.

CAP-based projects benefit from a primary focus on the domain. Instead of delving into overly technical disciplines, CAP focuses on accelerated development and safeguarding investments in a world of rapidly changing cloud technologies.

Here are some of the benefits that come with the SAP Cloud Application Programming Model (CAP):

- Built-in best practices
- Support for Visual Studio Code and SAP Business Application Studio tools
- Safeguarded application development investments
- No runtimes lock-in
- Reuse and integration of SAP BTP application services

- Latest UX and themes

See [SAP Cloud Application Programming Model \(CAP\)](#) in [SAP BTP Developer's Guide](#).

## 2.9.4 Continuous Integration and Delivery (CI/CD)

Configure and run predefined continuous integration and delivery (CI/CD) pipelines that automatically build, test, and deploy your code changes to speed up your development and delivery cycles.

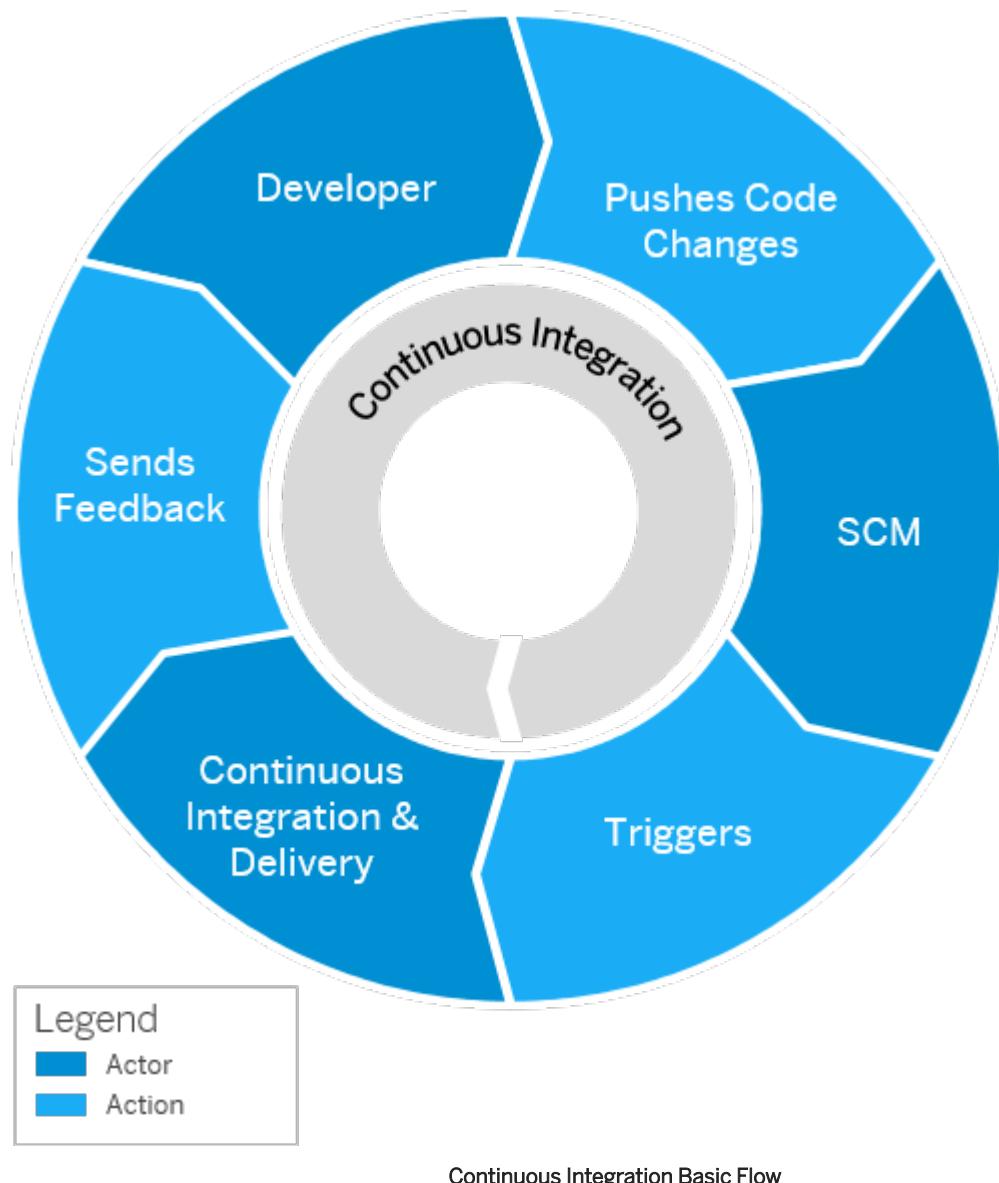
### Note

For links to all SAP solutions for CI/CD, blog posts, presentations, and tutorials, have a look at our [Continuous Integration and Delivery by SAP](#) overview.

**Continuous integration (CI)** describes a software development process, in which various team members integrate their contributions frequently into a single main line. Before each integration, the changes are verified through builds and automated testing. Thereby, you can detect errors as quickly as possible and prevent integration problems before completing the development.

The **continuous delivery (CD)** concept expands on the one of continuous integration. It adds the aspect that any change that has successfully passed the tests is immediately ready to be deployed to production, both from a technical and a qualitative point of view.

The following graphic shows the basic flow for continuous integration and delivery:



For more information about the continuous integration and continuous delivery concepts, see [What Are Continuous Integration and Continuous Delivery?](#).

## Use

Depending on your use case, you can choose between different CI/CD pipelines to help you implement continuous integration and delivery in your software development.

SAP Continuous Integration and Delivery lets you configure and run predefined pipelines for the development of the following applications:

- [SAP Cloud Application Programming Model](#)

Configure a CI/CD pipeline for the development of applications that follow the SAP Cloud Application Programming Model in the Cloud Foundry runtime.

- [Configure an SAP Fiori in the Cloud Foundry Environment](#)

Configure a CI/CD pipeline for the development of SAPUI5/SAP Fiori applications in the Cloud Foundry runtime.

- [Configure an SAP Fiori for the ABAP Platform](#)

Configure a CI/CD pipeline for the development of SAPUI5/SAP Fiori applications for the ABAP platform.

- [Configure an SAP Integration Suite Artifacts](#)

Configure a CI/CD pipeline for the development of SAP Cloud Integration artifacts in the Cloud Foundry runtime.

- [Configure a Container-Based Applications](#)

Configure a CI/CD pipeline for the development of container-based applications.

To learn more about the CI/CD pipelines supported by SAP Continuous Integration and Delivery and the stages each pipeline can comprise, see [Supported Pipelines](#).

## Get Started with CI/CD

SAP Continuous Integration and Delivery provides an easy, UI-guided way to set up the service and configure and run your pipelines, without hosting your own Jenkins instance.

To set up SAP Continuous Integration and Delivery:

1. Enable the service in the SAP BTP cockpit.
2. Assign either the *Administrator* or *Developer* role to your user.
3. Enable the API usage to connect SAP Continuous Integration and Delivery to other services, if necessary.

To configure SAP Continuous Integration and Delivery:

### ⓘ Note

Only administrators of SAP Continuous Integration and Delivery can configure the service.

1. Configure credentials for connecting SAP Continuous Integration and Delivery to other services (for example, GitHub, GitLab, Bitbucket Server, or Azure Repos to clone your sources, and SAP BTP to deploy your built application).
2. Add your repository.

Now you can create and modify your CI/CD jobs and monitor their outcome. If you want to automate your builds, you can configure a webhook between your repository and the service. You can create and modify timed triggers for your jobs, if necessary.

For more information, see [Initial Setup and Configuration](#).

## Automate the Provisioning of Resources Needed to Create a Kyma Cluster

Integrate the Terraform module, which uses the Terraform provider for SAP BTP, into CI/CD pipelines, so you can automate the provisioning of resources needed to create a Kyma cluster, deploy applications, run tests,

and tear down the resources when they are no longer required. As there is no need to have resources running continuously, this approach reduces operational costs.

See [Terraforming Kyma Runtimes](#).

## Learn and Get Certified

Depending on your learning goals and level of expertise, you can choose from the following offerings:

- **Efficient DevOps with SAP** 

This openSAP course introduces general DevOps approaches and key principles. Learn about the basic CI/CD principles (week 1) and how to deliver cloud applications using CI/CD (week 3).

- **Continuous Integration and Delivery Introduction Guide**

The CI/CD Introduction Guide provides you with basic knowledge for setting up and implementing continuous integration and delivery processes. It gives an overview of the concepts and principles of CI/CD, explains both procedures and their relation, and helps you plan your own CI/CD process.

## 2.9.5 APIs

Discover and consume APIs to manage, build, and extend the core capabilities of SAP BTP.

An Application Programming Interface or API is an **interface provided by an application for interacting with other applications**. APIs specify how software programs are able to exchange information with each other, even if designed and run by different organizations. They facilitate interaction by selectively exposing certain functionality, allowing different apps, websites, and devices to communicate effectively with each other. More importantly, APIs enable businesses to reach beyond regular business channels and share data, content, and services directly to both B2B (business to business) and B2C (business to consumer) clients, making UI development easy.

## API Offerings on SAP BTP

SAP BTP enables you to consume APIs and publish your own ones through the following offerings:

Offering	Description
<a href="#">SAP BTP on SAP Business Accelerator Hub</a> 	<p>The SAP Business Accelerator Hub provides you with one central repository for browsing and accessing APIs from SAP and selected partners. Test APIs and try out mock data in your systems.</p> <p>It is also the official place where REST and OData REST API references are published.</p>

Offering	Description
<a href="#">API Management Capability within SAP Integration Suite</a>	With the API Management capability, you can build, manage, publish, and monetize your own APIs within one secure and scalable environment.
<b>Software Development Kits (SDKs)</b>  For an overview of the available SDKs, see <a href="#">Tools [page 121]</a> .	The SDKs available for SAP BTP offer APIs to, for example, accelerate enterprise app development.

## Related Information

[SAP API Style Guide](#)

## 2.9.6 Cloud Management Tools — Feature Set Overview

Cloud management tools represent the group of technologies designed for managing SAP BTP.

We're currently renovating and adding core functionalities to SAP BTP. As part of this process, we're upgrading enterprise accounts from the existing cloud management tools feature set A to the renovated cloud management tools feature set B. We're doing this upgrade as a phased rollout, so cloud management tools feature sets A and B will coexist for some time. For more information about the account upgrade process and frequently-asked questions, see [3027721](#).

### How can I know which cloud management tools feature set I'm using?

There's an easy way to check if you're currently using cloud management tools feature set A or B. Access the SAP BTP cockpit and choose your username from the top right-hand corner of the screen. From the menu, select [① About](#) to get information about the cloud management tools feature set you're using.



## Access the SAP BTP cockpit

Feature sets A and B don't share the same cockpit. So in your feature set A cockpit, you'll only see feature set A global accounts, while in your feature set B cockpit, you'll only see your feature set B global accounts.

When using cloud management tools feature set A: Choose <https://account.eu1.hana.ondemand.com> to access the SAP BTP cockpit.

When using cloud management tools feature set B: Choose <https://cockpit.btp.cloud.sap> to access the cockpit. Depending on your own geo location this URL will redirect you to the closest regional Cockpit URL.

For more information, see [Access the Cockpit \[page 2141\]](#).

## What are the differences between the two cloud management tools feature sets?

New/Changed Features and Behaviors	Feature Set A	Feature Set B
Directories — NEW	<i>Not applicable</i>	<p>Summary:</p> <ul style="list-style-type: none"><li>• Group and filter directories and subaccounts</li><li>• Monitor usage and costs for contracts that use the consumption-based commercial model</li></ul> <p>With <b>directories</b>, you can organize and manage your subaccounts according to your technical and business needs.</p> <p>A directory can contain directories and subaccounts to create a hierarchy. Using directories to group other directories and subaccounts is optional - you can still create subaccounts directly under your global account.</p> <p>See:</p> <p><a href="#">Directories [page 96]</a></p> <p><a href="#">Create a Directory [page 2166]</a></p> <p><a href="#">Manage the Account Explorer Hierarchy [page 2168]</a></p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
Labels — NEW	<i>Not applicable</i>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Set labels for categorization and identification purposes</li> <li>• Additional filtering options</li> </ul> <p>Labels allow you to categorize your directories, subaccounts, multitenant application subscriptions, service instances, and environment instances so that you identify them more easily within your global account. For example, the <a href="#">Account Explorer</a> and <a href="#">Instances and Subscriptions</a> pages in the cockpit allow you search by label name or value.</p> <p>Labels are user-defined so you can apply them as you wish according to your own business and technical needs.</p> <p>You can manage labels for the supported entities using the SAP BTP cockpit, command line interface (btp CLI), or REST APIs.</p> <p>See <a href="#">Labels [page 98]</a>.</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
APIs for SAP BTP — NEW	<i>Not applicable</i>	<p>Summary:</p> <p>Discover and consume REST APIs to manage, build, and extend the cloud operation capabilities of SAP BTP. For example:</p> <ul style="list-style-type: none"> <li>• Manage global accounts, directories, and subaccounts</li> <li>• Assign entitlements for services and applications to directories and subaccounts</li> <li>• Manage the provisioning of environment instances and get information relating to provisioned services</li> <li>• Manage subaccount subscriptions to multitenant applications</li> <li>• Generate reports based on the resource and cost consumption within your accounts.</li> </ul> <p>See <a href="#">Account Administration Using APIs of the SAP Cloud Management Service [page 2378]</a> and <a href="#">Monitoring Usage Information Using APIs of the SAP Usage Data Management Service [page 2409]</a>.</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
SAP BTP command line interface ( <b>btp CLI</b> ) — NEW	<p><i>Not applicable</i></p>	<p>Summary:</p> <p>Use the btp CLI for convenient account management tasks on the command line and to automate these procedures. For example:</p> <ul style="list-style-type: none"> <li>• Manage global accounts, directories, and subaccounts</li> <li>• Set entitlements</li> <li>• Work with environments</li> <li>• Work with multitenant applications</li> <li>• Manage users and their authorizations</li> </ul> <p>To log in to a global account, you need the CLI server URL <code>https://cpcli.cf.eu10.hana.ondemand.com</code> and the global account subdomain. Only global accounts on feature set B have global account subdomains (see cockpit).</p> <p>See:</p> <p><a href="#">Account Administration Using the SAP BTP Command Line Interface (btp CLI) [page 2321]</a></p> <p><a href="#">btp CLI Command Reference</a></p> <p><a href="#">Setting Up a Global Account via the Command Line [page 162]</a></p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
<b>Global account navigation — CHANGED</b>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• "Home" scope outside the global account in the cockpit</li> <li>• <i>Global Accounts</i> page in the cockpit</li> <li>• Switching between global accounts via breadcrumbs (second element)</li> <li>• <i>Overview</i> page is the first in the global account scope</li> </ul> <p>With feature set A, your cockpit contains a number of general views <b>outside</b> of the global account scope. These include a <i>Global Accounts</i> page, where you can find all your global accounts listed as tiles, and from where you can navigate to your desired global account.</p> <p>Since it has several views outside of the global account scope, your cockpit contains an additional "home" scope, which is represented by the first element in the breadcrumbs. Your global account is therefore represented by the second element in the breadcrumbs. You can also use that second element to navigate from one global account to another.</p> <p>Once you're in a global account, the first page you see is the global account overview page. To navigate to a sub-account, you have to navigate to the <i>Subaccounts</i> page from the left hand-side navigation.</p> <p>See <a href="#">Navigate in the Cockpit [page 2142]</a></p>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Global accounts selection dialog displayed before entering the cockpit</li> <li>• Option to set a default global account</li> <li>• "Home" scope removed from the cockpit</li> <li>• Switching between global accounts via the selection dialog or breadcrumbs (first element)</li> <li>• <i>Subaccounts</i> page is the first in the global account scope</li> </ul> <p>With feature set B, there's no scope beyond the global account in the cockpit anymore. Therefore, after logging on to the cockpit and before actually entering it, if you have more than one global account, you're asked to choose which of the available cloud management tools feature set B global accounts you want to enter. Only feature set B global accounts are visible here. This is done via a global account selection dialog, where you also have the option to remember your selection. Doing that sets the global account you chose as default, so that next time you access the cockpit you're automatically taken to your default global account instead of seeing the selection dialog.</p> <p>As the cockpit doesn't have a "home" scope anymore, the global account becomes the outermost scope and is therefore represented by the first element in the breadcrumbs. You can still navigate from one global account to another either by using the breadcrumbs, or by choosing <i>Switch Global Accounts</i> to launch the global account selection dialog, where you can also modify or remove your default global account.</p> <p>Once you're in a global account, the first page you see is the <i>Subaccounts</i> page, from where you can directly nav-</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
		<p>igate to your subaccount — one less click needed. Usage and cost information for your global account is displayed in the global account's <i>Usage Analytics</i> page.</p> <p>See <a href="#">Navigate in the Cockpit [page 2142]</a>.</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
<b>Entitlements — CHANGED</b>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Only manage entitlements for services</li> <li>• Assign entitlements to subaccounts individually</li> <li>• <i>Service Assignments</i> view</li> </ul> <p>With feature set A, entitlements only apply to services and you have two views for them: <i>Subaccount Assignments</i>, where you can assign or edit entitlements to individual subaccounts, and <i>Service Assignments</i>, which is a read-only view displaying the distribution of your available services across subaccounts.</p> <p>See:</p> <p><a href="#">Entitlements and Quotas [page 100]</a></p> <p><a href="#">Managing Entitlements and Quotas Using the Cockpit [page 2185]</a></p> <p><a href="#">Configure Entitlements and Quotas for Subaccounts [page 2193]</a></p>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Manage entitlements for both services and multitenant applications</li> <li>• Assign entitlements to subaccounts individually</li> <li>• Assign entitlements to directories</li> <li>• Auto-assign entitlements to all new subaccounts in a directory</li> <li>• One single view for your entitlements, no <i>Service Assignments</i> page</li> </ul> <p>With feature set B, you manage entitlements for both services and multitenant applications. Since you have directories as a way to group your subaccounts, you can also assign entitlements to a directory and choose the option to automatically assign a certain amount of quota to each subaccount added to that directory in the future (as long as it doesn't exceed the quota of that directory).</p> <p>This means that you can more efficiently assign quota to multiple subaccounts that should have the same entitlements.</p> <p>In addition, you no longer have the <i>Service Assignments</i> view with feature set B.</p> <p>See:</p> <p><a href="#">Entitlements and Quotas [page 100]</a></p> <p><a href="#">Managing Entitlements and Quotas Using the Cockpit [page 2185]</a></p> <p><a href="#">Configure Entitlements and Quotas for Directories [page 2188]</a></p> <p><a href="#">Subscribe to Multitenant Applications Using the Cockpit [page 2198]</a></p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
<b>Global account security — CHANGED</b>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Global account membership determines if a user is global account administrator or not.</li> <li>• Members of subaccounts have view-only access to their global accounts.</li> </ul> <p>With feature set A, the <a href="#">Members</a> tab determines which users are global administrators. These users can assign or remove global administrator membership to other users.</p> <p>See:</p> <p><a href="#">User and Member Management [page 104]</a></p> <p><a href="#">Add Members to Your Global Account [page 2146]</a></p> <p><a href="#">Impact of Upgrading from Feature Set A to Feature Set B on User and Account Management [page 116]</a></p>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• Global account membership is determined by the assignment of a role collection.</li> <li>• Predefined role collections for global accounts define full and read-only access.</li> <li>• You can define your own role collections with the authorizations delivered by SAP.</li> </ul> <p>With feature set B, you have a fine-grained authorization concept for the management of global accounts. We deliver a set of role collections for the management of global accounts. If these role collections don't match your needs, you can configure your own role collections using the authorizations we supply. For example, you have control over which global account users can create subaccounts and which can't.</p> <p>The Authorization and Trust Management (XSUAA) service is responsible for access management for global accounts. This service is the same service that performs access management at the subaccount level, although it's a different instance of the service.</p> <p>In the cockpit, a new <a href="#">Security</a> tab provides access to management functions for role collections and user assignment.</p> <p>With feature set B, global account users are identified by their e-mail address and not their user ID.</p> <p>See:</p> <p><a href="#">User and Member Management [page 104]</a></p> <p><a href="#">Role Collections and Roles in Global Accounts, Directories, and Subaccounts [page 107]</a></p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
		<a href="#">Default Role Collections of SAP BTP [page 3090]</a>
<b>Subaccount security — CHANGED</b>	<p>Summary:</p> <ul style="list-style-type: none"> <li>Subaccount administrators are determined by assignment under security administrators.</li> <li>Subaccount members are determined by assignment of roles in the Cloud Foundry org under the <a href="#">Members</a> tab.</li> </ul> <p>With feature set A, you define administrators in the cockpit under the <a href="#">Security</a> tab with the <a href="#">Administrators</a> menu item.</p> <p>See:</p> <p><a href="#">User and Member Management [page 104]</a></p> <p><a href="#">Impact of Upgrading from Feature Set A to Feature Set B on User and Account Management [page 116]</a></p>	<p>Summary:</p> <ul style="list-style-type: none"> <li>Subaccount administrators and members are determined by the assignment of role collections.</li> <li>Predefined role collections for subaccounts define the level of access.</li> <li>Provision subaccount members with SCIM APIs</li> <li>Membership in subaccounts and any environments, such as the Cloud Foundry org, are controlled by separate authorizations.</li> </ul> <p>With feature set B, you have a fine-grained authorization concept for the management of subaccounts. We deliver a set of role collections for the management of subaccounts. If these role collections don't match your needs, you can configure your own role collections using the authorizations we supply.</p> <p>Access to environments, such as a Cloud Foundry org, is semi-independent from subaccount membership. A subaccount member isn't necessarily a member of an environment, but a member of an environment is a member of its subaccount.</p> <p>See:</p> <p><a href="#">User and Member Management [page 104]</a></p> <p><a href="#">Role Collections and Roles in Global Accounts, Directories, and Subaccounts [page 107]</a></p> <p><a href="#">Add Members to Your Subaccount [page 2177]</a></p> <p><a href="#">Default Role Collections of SAP BTP [page 3090]</a></p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
<b>Custom Identity Provider for Platform Users — CHANGED</b>	<p>Summary:</p> <p>Custom identity providers enable you to integrate platform and business users from SAP Cloud Identity Services - Identity Authentication or your own corporate identity provider.</p> <p>See: <a href="#">Trust and Federation with Identity Providers [page 2204]</a></p> <p><a href="#">Impact of Upgrading from Feature Set A to Feature Set B on User and Account Management [page 116]</a></p>	<p>Summary:</p> <p>Custom identity providers enable you to integrate platform and business users from SAP Cloud Identity Services - Identity Authentication or your own corporate identity provider.</p> <p>See: <a href="#">Trust and Federation with Identity Providers [page 2204]</a></p> <p><a href="#">Restrictions When Using Custom Identity Providers for Platform Users [page 2243]</a></p>
<b>(Trial Only) Automatic Setup of Trial Account — CHANGED</b>	<p>Not applicable</p>	<p>Summary:</p> <ul style="list-style-type: none"> <li>Your trial account is automatically set up for you after you choose <a href="#">Enter Your Trial Account</a> from the trial homepage in the cockpit for the first time.</li> <li>You get a global account with a subaccount called <i>trial</i>, which in turn contains an org and a space called <i>dev</i>.</li> <li>All entitlements are assigned to the subaccount that is provisioned automatically.</li> </ul> <p>With feature set B, you can access the trial homepage in the cockpit before your trial account is set up and ready to use. This means that you can launch a starter scenario or guided tour before you have the global account, subaccount, org, space and entitlements in place. You trigger the automatic creation when you first choose <a href="#">Enter Your Trial Account</a> from the trial homepage.</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
(Trial Only) Trial Account Extension - CHANGED	<i>Not applicable</i>	<p>Summary:</p> <ul style="list-style-type: none"> <li>• You cannot access a suspended trial global account.</li> <li>• You can still see the counter with the remaining number of days in the same place, but you cannot extend your trial from there.</li> <li>• You extend your trial account from a dialog similar to the global account selection dialog described previously in this table under global account navigation.</li> </ul> <p>With feature set B, it's not possible anymore to access suspended trial global accounts. Before your trial interval expires, you can still see the counter with the remaining number of days in the same place. However, once your trial interval expires and you try to access it, you will instead be prompted by a dialog asking you to extend your trial first.</p> <p><b> ⓘ Note</b></p> <p>The overall trial period is 90 days and is divided in intervals.</p> <p>If you don't log in to your account for 30 days or more, your account will be suspended. During suspension, your applications may be stopped and you won't be able to access them. However, your data will not be deleted yet. You can unsuspend your account as long as there are days left in your trial period.</p> <p>When your 90-day trial period is finished, your account will be deleted, and you will no longer be able to access your data. You can then set up a new trial account.</p>

New/Changed Features and Behaviors	Feature Set A	Feature Set B
(Trial Only) Deletion of SAP BTP Trial - NEW	<i>Not applicable</i>	<p>Summary:</p> <ul style="list-style-type: none"> <li>Possible to easily delete only your SAP BTP trial account.</li> </ul> <p>With feature set B, you can delete your SAP BTP trial account. Simply navigate into your trial global account by choosing <a href="#">Enter Your Trial Account</a> on the trial home page. Once you're on the <a href="#">Subaccounts</a> page of your global account, you will see a new button giving you the option to delete your trial account.</p>

#### ⓘ Note

Cloud management tools feature set B applies also to the Neo environment. For more information about the scope offered with the enhanced capabilities, see [Working with Cloud Management Tools Feature Set B in the Neo Environment](#).

## 2.9.7 Prerequisites and Restrictions

Find a list of the product prerequisites and restrictions for SAP BTP.

### General Constraints

- For information on constraints and default settings to consider when you deploy an application in the Cloud Foundry environment, see [http://docs.cloudfoundry.org/devguide/deploy-apps/large-app-deploy.html#limits\\_table](http://docs.cloudfoundry.org/devguide/deploy-apps/large-app-deploy.html#limits_table).
- SAP BTP exposes applications only via HTTPS. For security reasons, applications can't be accessed via HTTP.

### SAP BTP Tools

- SAP BTP Tools for Java and SDK have been tested with Java 7, and Java 8.
- SAP BTP Tools for Java and SDK run in many operating environments with Java 7, and Java 8 that are supported by Eclipse. However, SAP doesn't systematically test all platforms.
- SAP BTP Tools for Java must be installed on Eclipse IDE for Java EE developers.

For the platform development tools, SDK, Cloud connector, and SAP JVM, see <https://tools.hana.ondemand.com/#cloud>.

## Browser Support

For a list of supported browsers for the SAP BTP cockpit, see [Feature Scope Description](#).

For a list of supported browsers for developing SAPUI5 applications, see [Browser and Platform Support](#).

To find out the browser support for a specific service, refer to the corresponding feature scope description.

For security reasons, SAP BTP doesn't support TLS1.1 and older, SSL 3.0 and older, and RC4-based cipher suites. Make sure that your browser supports at least TLS1.2 and modern ciphers (for example, AES).

# 3 Getting Started

Once you're familiar with the basic concepts of SAP BTP, you can start your first workflows or check out further resources that help you get started quickly.

## Onboarding Guides

The guide [SAP BTP Administrator's Guide](#) helps you plan your development projects on SAP BTP from onboarding to SAP Cloud Identity Services, through setting up the correct organizational structure, to creating an account and security model, to developing and operating applications.

The guide [SAP BTP Developer's Guide](#) helps you define the correct methodologies and tools for your development project.

## Getting Started Workflows

Learn how to get a global account and get started with a trial or an enterprise account on SAP BTP. A trial account lets you try out SAP BTP for free, and within an enterprise account, you can use so-called free tier service plans for free. See [Trial Accounts and Free Tier \[page 82\]](#).

Depending on your use case, follow the appropriate workflow:

- [Getting a Global Account \[page 144\]](#)
- [Getting Started in the Cloud Foundry Environment \[page 147\]](#)
- [Getting Started in the ABAP Environment \[page 165\]](#)
- [Getting Started in the Kyma Environment \[page 223\]](#)

## Additional Resources

The SAP Developer Center provides further resources to get you started:

- [SAP Discovery Center!\[\]\(706a40097f49e54c461e398b5f4f8844\_img.jpg\)](#)
- [SAP BTP in SAP Developer Center!\[\]\(9717baf2235f4f40564d43bbc589d7d3\_img.jpg\)](#)
- [Start Developing on SAP BTP Tutorials!\[\]\(d3ade5078a3f1e9f532f1e7e33f95217\_img.jpg\)](#)

## Related Information

[SAP BTP Administrator's Guide](#)

## 3.1 Getting a Global Account

SAP BTP offers two types of global accounts: Trial accounts (completely free of charge) and enterprise accounts. Within an enterprise account, you can use both free and paid plans.

A **global account** is the realization of a contract you or your company has made with SAP. A global account is used to manage subaccounts, members, entitlements and quotas. You receive entitlements and quotas to use platform resources per global account and then distribute the entitlements and quotas to the subaccount for actual consumption. There are two types of commercial models for global accounts: consumption-based model and subscription-based model. See [Commercial Models \[page 86\]](#)

- [Trial Accounts and Free Tier \[page 82\]](#)
- [Enterprise Accounts \[page 84\]](#)
- [Try Out SAP BTP for Free \[page 144\]](#)
- [Get an Enterprise Account \[page 145\]](#)

### 3.1.1 Try Out SAP BTP for Free

There are two ways to try out SAP BTP services and applications for free.

#### Context

A **free trial** account is intended for individuals, for example students, who want to explore SAP BTP for free. It is an isolated account and subaccount, with preallocated service entitlements that can be used for a limited number of days.

**Free tier** is not an account type, but it's a set of free services that can be used within an enterprise account.

To learn which is best for your needs, see the [Set Up an SAP BTP Account for Tutorials](#) group in the SAP Developer Center.

## 3.1.2 Get an Enterprise Account

To use an enterprise account, you can either purchase a customer account, join the partner program to purchase a partner account, or self-register for an enterprise account to try out free tier service plans.

For more information about the scope of our enterprise offering, see [Enterprise Accounts \[page 84\]](#). For more information about free tier within an enterprise account, see [Trial Accounts and Free Tier \[page 82\]](#) and [Using Free Service Plans \[page 91\]](#).

### Related Information

[Get an Account on SAP BTP to Try Out Free Tier Service Plans](#)

### 3.1.2.1 Sign up for a Customer Account

A customer account is an enterprise account that allows you to host productive, business-critical applications with 24x7 support.

When you want to purchase a customer account, you can select from a set of predefined packages. For information about service availability, prices, and estimators, see <https://www.sap.com/products/technology-platform/pricing.html>. You can also view the service catalog via the [SAP Discovery Center](#). Contact us on [SAP BTP](#) or via an SAP sales representative.

### Free Tier

If you want to try out services for free, with the option of easily upgrading them later, you can get an enterprise account and use free tier service plans only. See [Get an Account on SAP BTP to Try Out Free Tier Service Plans](#). When you sign up for a global account to try out free tier service plans, you need to select either an existing global account of your company, or create a new global account. If you choose an existing global account, make sure to contact the global account admin of this global account, as they'll receive the communication emails. In this case, we recommend to have them add you as Global Account Administrator. See [Assign Users to Role Collections \[page 2278\]](#) and [SAP BTP Onboarding Resource Center](#).

You can upgrade and refine your resources later on. You can also contact your SAP sales representative and opt for a configuration, tailored to your needs.

### Onboarding

After you have purchased a new customer account, you will receive an email confirming the provisioning of resources from the platform services team and a second email with the URL and login ID to obtain access. Note, only the person who receives the access email has initial access to SAP BTP.

If you chose an add-on to an existing global account, the initial access email will not be sent to you. You will need to contact the Global Account Administrator from the initial BTP order. If you're unsure who this person is, please contact your SAP Sales representative. If you need to add a new global admin because the person with initial access is no longer available to grant you access, then you can contact the SAP Support Portal, by creating a ticket at component BC-NEO-CIS-OPS.

→ Tip

Check out the [Live Onboarding Webinars](#).

## Getting a Customer Account in China

If you are located in China and want to buy a global account on SAP BTP, you need to contact an SAP sales representative: <https://www.sap.cn/registration/contact.html>.

## Related Information

[Commercial Models \[page 86\]](#)

[Using Free Service Plans \[page 91\]](#)

[Access the Cockpit \[page 2141\]](#)

### 3.1.2.2 Join the Partner Program

A partner account is an enterprise account that enables you to build applications and to sell them to your customers.

To become a partner, you need to fill in an application form and then sign your partner contract. You will be assigned to a partner account with the respective resources. To apply for the partner program, visit [https://partneredge.sap.com/content/partnerregistration/en\\_us/registration.html?partnertype=BLD&engagement=0002&build=1](https://partneredge.sap.com/content/partnerregistration/en_us/registration.html?partnertype=BLD&engagement=0002&build=1). You will receive a welcome mail with further information afterwards.

General information about the partner program is available on <https://www.sap.com/partners/partner-program.html>.

## Next Steps

- [Getting Started in the Cloud Foundry Environment \[page 147\]](#)
- [Getting Started in the ABAP Environment \[page 165\]](#)
- [Getting Started in the Kyma Environment \[page 223\]](#)

## 3.2 Getting Started in the Cloud Foundry Environment

Get onboarded in the Cloud Foundry environment of SAP BTP. Follow the workflows for trial or customer accounts or subscribe to business applications.

[Getting Started with a Trial Account in the Cloud Foundry Environment \[page 147\]](#)

Quickly get started with a trial account.

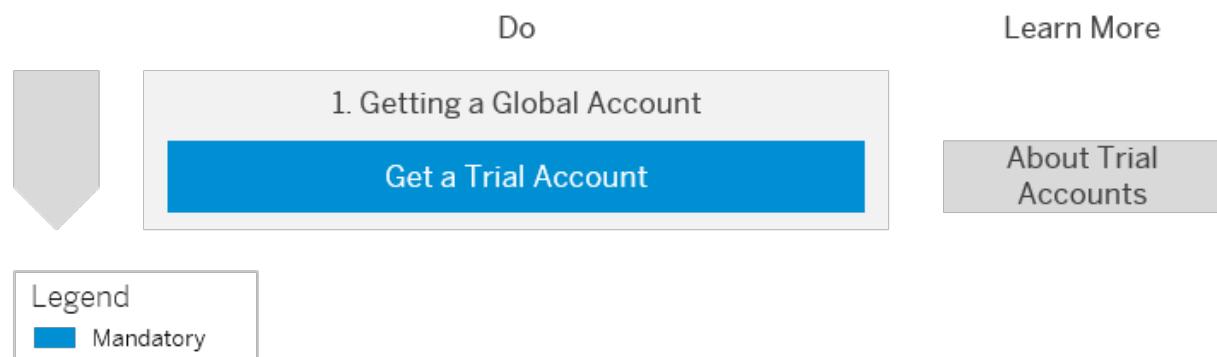
[Getting Started with an Enterprise Account in the Cloud Foundry Environment \[page 158\]](#)

Quickly get started with an enterprise account in the Cloud Foundry Environment.

### 3.2.1 Getting Started with a Trial Account in the Cloud Foundry Environment

Quickly get started with a trial account.

#### 1. Getting a Global Account

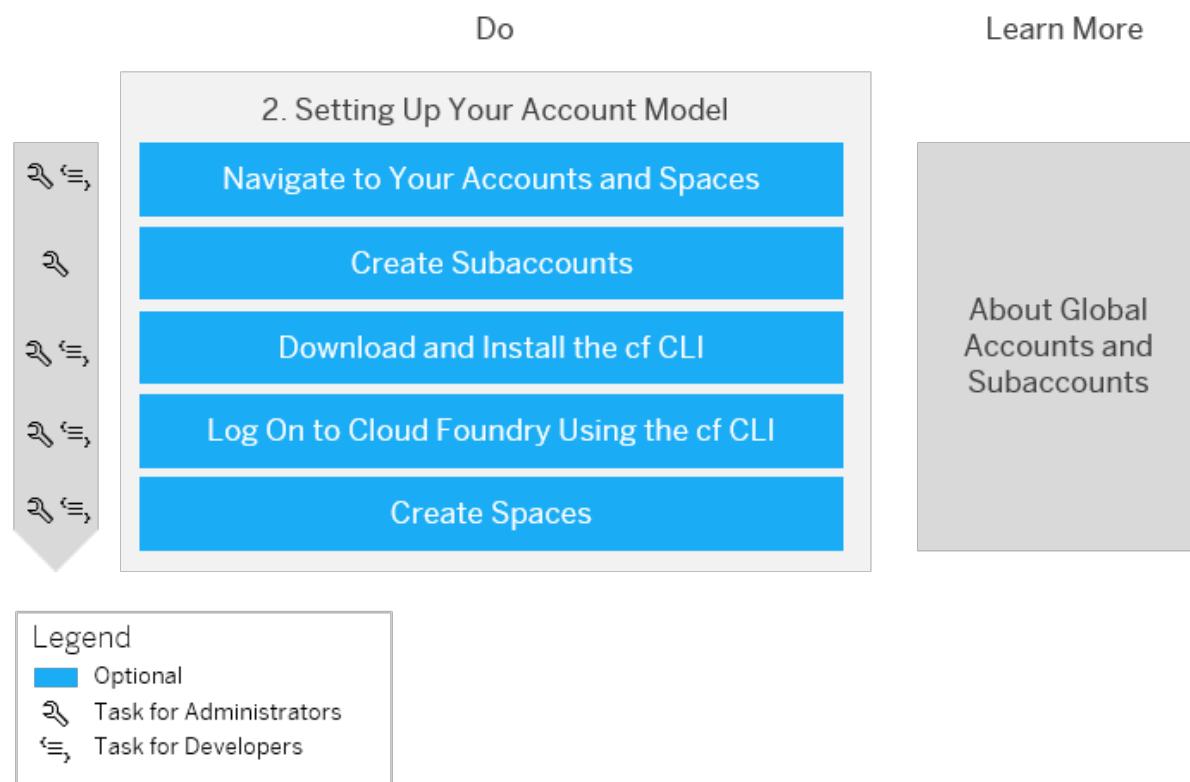


- [Try Out SAP BTP for Free \[page 144\]](#)
- [Trial Accounts and Free Tier \[page 82\]](#)

Before you begin, sign up for a free trial account. See [Get a Free Trial Account \[page 144\]](#). For more information about the scope of our trial offering, see [Trial Accounts and Free Tier \[page 82\]](#).

If you want to familiarize yourself with the Cloud Foundry environment, see [Cloud Foundry Environment \[page 49\]](#).

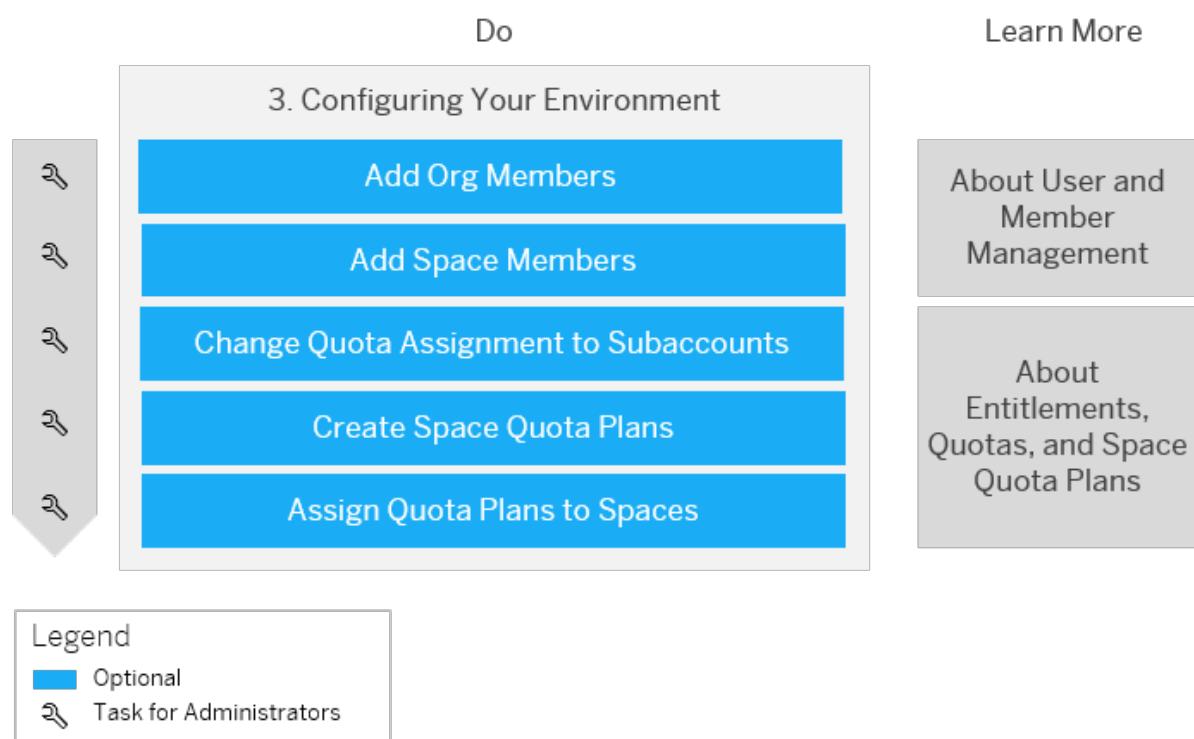
## 2. Setting Up Your Account Model



- [Navigate to Orgs and Spaces \[page 2433\]](#)
  - [Create a Subaccount \[page 2173\]](#)
  - [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
  - [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)
  - [Create Spaces \[page 2441\]](#)
  - [Account Model \[page 94\]](#)
1. When you register for a trial account, a subaccount and a space are created for you. You can create additional subaccounts and spaces, thereby further breaking down your account model and structuring it according to your development scenario, but first it's important you understand how to navigate to your accounts and spaces using the SAP BTP cockpit. See [Navigate to Orgs and Spaces \[page 2433\]](#).
  2. If you like, create further subaccounts. See [Create a Subaccount \[page 2173\]](#). You can also download and use the CLI for SAP BTP to create new subaccounts. See [Download and Start Using the btp CLI Client \[page 2323\]](#) and [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#).
  3. If you haven't done so already, now is a good time to download and install the Cloud Foundry Command Line Interface (cf CLI). This tool allows you to administer and configure your environment, enable services, and deploy applications. See [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#). But don't worry, you can also perform all the necessary task using the SAP BTP cockpit, which you don't need to install.

- If you'd like to use the cf CLI, log on to your environment. See [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- If you like, create further spaces. See [Create Spaces \[page 2441\]](#). If you want to learn more about subaccounts, orgs, and spaces, and how they relate to each other, see [Account Model \[page 94\]](#).

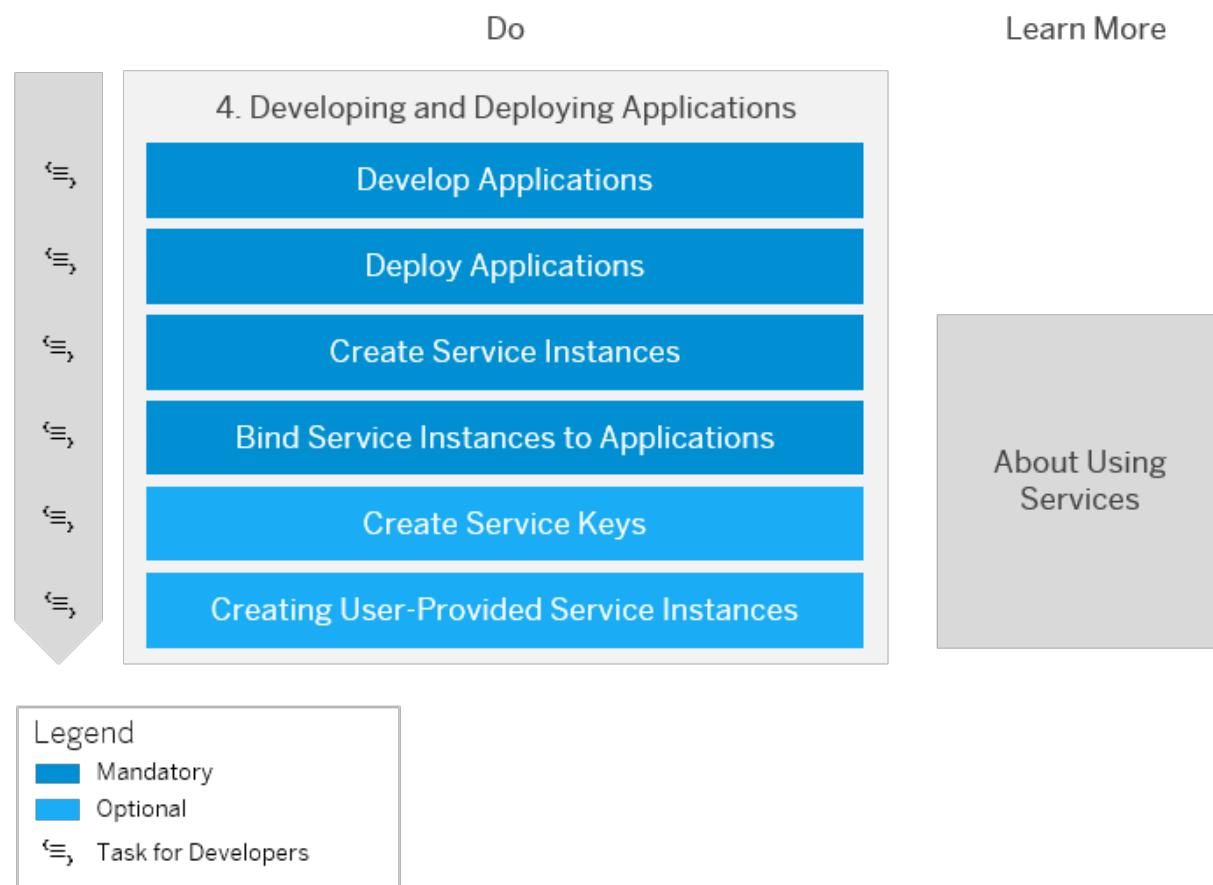
### 3. Configuring Your Environment



- [Add Org Members \[page 2437\]](#)
  - [Add Space Members \[page 2444\]](#)
  - [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)
  - [Create Space Quotas \[page 2449\]](#)
  - [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#)
  - [User and Member Management \[page 104\]](#)
  - [Entitlements and Quotas \[page 100\]](#)
- Now that you've set up your account model, it's time to think about member management. You can add members at different levels. For example, you can add members at the org level. See [Add Org Members \[page 2437\]](#). For more information about the roles that are available on the different levels, see [User and Member Management \[page 104\]](#).
  - You can also add members at the space level. See [Add Space Members \[page 2444\]](#).

- In a trial account, quotas are automatically assigned to your subaccounts, but you can change that assignment. See [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#). To learn more about entitlements and quotas, see [Entitlements and Quotas \[page 100\]](#).
- You can also assign quotas to different spaces within a subaccount. To do so, first create a space quota plan. See [Create Space Quotas \[page 2449\]](#) or [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#).
- Then assign the quota plan to your space. See [Assign Space Quotas to Spaces \[page 2450\]](#) or [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 2488\]](#).

## 4. Developing and Deploying Applications



- [Development \[page 231\]](#)
- [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#)
- [Creating Service Instances \[page 250\]](#)
- [Binding Service Instances to Applications \[page 253\]](#)
- [Creating Service Keys \[page 255\]](#)

- [About Services \[page 249\]](#)
  - [Creating User-Provided Service Instances \[page 252\]](#)
1. Develop your application. Check out the Developer Guide for tutorials and more information. See [Development \[page 231\]](#).
  2. Deploy your application. See [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#).
  3. Integrate your application with a service. To do so, first create a service instance. See [Creating Service Instances \[page 250\]](#).
  4. Bind the service instance to your application. See [Binding Service Instances to Applications \[page 253\]](#).
  5. Alternatively, you can also create and use service keys. See [Creating Service Keys \[page 255\]](#). For more information on using services and creating service keys, see [About Services \[page 249\]](#).
  6. You can also create instances of user-provided services. See [Creating User-Provided Service Instances \[page 252\]](#).

→ Tip

Also check out the tutorial [Create Your First App on Cloud Foundry](#) to see how you can deploy a pre-bundled set of artifacts using the SAP BTP cockpit, access the app from your web browser, and create an instance of a service available on Cloud Foundry and bind it to your app.

## Related Information

[Cloud Foundry Environment \[page 49\]](#)

[Trial Accounts and Free Tier \[page 82\]](#)

[Account Model \[page 94\]](#)

[Entitlements and Quotas \[page 100\]](#)

[User and Member Management \[page 104\]](#)

[About Services \[page 249\]](#)

## 3.2.1.1 Setting Up Your Trial Account

Your trial account is set up automatically, so that you can start using it right away. However, if one or more of the automatic steps fail, you can also finalize the setup manually, by following the steps below.

### 3.2.1.1.1 Create Your Trial Subaccount

The first thing that is needed in the setup of your trial account is the creation of a subaccount. If this step was successful in the SAP BTP cockpit, you can directly skip to the next section.

#### Procedure

1. Navigate into your global account by choosing [Enter Your Trial Account](#).
2. Choose [New Subaccount](#).
3. Configure it as follows:

Field	Input
Display Name	<code>trial</code>
Description	(Optional) Any text that describes your subaccount
Provider	Desired infrastructure provider
Region	Desired region
Subdomain	<code>&lt;your_id&gt;trial</code> Example: <code>P0123456789trial</code>
[Advanced] Enable beta features	(Optional) Enables the use of beta services and applications.
[Advanced] Labels	(Optional) You can assign labels to help identify and organize the subaccounts in your global account. For example, you can filter subaccounts by custom property in the <a href="#">Account Explorer</a> page. See <a href="#">Labels [page 98]</a> .

#### Results

You have successfully set up your trial subaccount.

### 3.2.1.1.2 Create Your Trial Org and Configure Entitlements

Once you have a subaccount (whether it was created automatically or you followed the steps described above), you need an org and entitlements.

#### Procedure

1. Navigate to your subaccount by choosing its tile.
2. Choose *Enable Cloud Foundry* to create your org.
3. Once your org is created, choose *Entitlements* from the left hand-side navigation.
4. Choose *Configure Entitlements* to enter edit mode.
5. Choose *Add Service Plans* and select all the service plans available for your subaccount.

#### ⓘ Note

To select a service plan, choose a service from the left and tick all the service plans that appear on the right. Do that for all services.

6. Once you've added all the service plans, you see them all in a table. Before you choose Save, for all the plans with numerical quota, choose + to increase the amount to the maximum (until the icon becomes disabled).
7. Finally, choose *Save* to save all your changes and exit edit mode.

#### Results

You now have an org and all the entitlements for your subaccount. The last thing you need is a space where you can use the services you've configured entitlements for and deploy applications.

### 3.2.1.1.3 Create Your Trial Space

#### Procedure

1. In your *trial* subaccount, navigate to ► *Cloud Foundry* ► *Spaces* ▶ using the left hand-side navigation.
2. Choose *Create Space* and name it **dev**.
3. Choose *Create*.

## Results

You now have your trial set up all done and ready to go.

To get some guidance on how you can get started, navigate back to your Trial Home by choosing the first item in your breadcrumbs at the top. There, you can find several guided tours to walk you through the basics of SAP Business Technology Platform and the cockpit, as well as some more complex starter scenarios.

### 3.2.1.2 Setting Up a Trial Account From the Command Line

You can use the command-line interface to set up a trial account. For all tasks on global account and subaccount levels, you use the **SAP BTP command-line interface (btp CLI)**. Once you've created a Cloud Foundry environment instance (a Cloud Foundry org), you use the Cloud Foundry CLI (cf CLI). This procedure works without the SAP BTP cockpit (except that you need the global account subdomain to log in, which you may have to look up in the cockpit).

For all tasks on global account and subaccount level, you can use the **the btp CLI** instead of the SAP BTP cockpit. Once you've created a Cloud Foundry environment instance (a Cloud Foundry org), use the Cloud Foundry CLI (cf CLI).

## Prerequisites

- You have created a trial account.
- You have downloaded and extracted the following command-line tools:
  - SAP BTP command-line interface (btp CLI). See [Download and Start Using the btp CLI Client \[page 2323\]](#)
  - Cloud Foundry CLI (cf CLI). See [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

## Procedure

### → Tip

In the btp CLI, you can view the command help of each command to get information about how to use the command, its syntax, and input parameters. See [Get Help \[page 2336\]](#).

Step No.	Task	Per-formed By	Do This	More Information
1.	Log in to your global account using the subdomain of your global account.	Global ac-count ad-ministrator or viewer	Run this command in the btp CLI: <code>btp login</code>	See <a href="#">Log in [page 2337]</a> .
2.	View details of your global account.	Global ac-count ad-min or viewer	Run this command in the btp CLI: <code>btp get accounts/global-account</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> .
4.	View all the regions that are available to your global account and subaccounts.	Global ac-count ad-min or viewer	Run this command in the btp CLI: <code>btp list accounts/available-region</code>	This command also provides information about the environments and infrastructure provider of each region.
5.	Create subaccounts in your global account.	Global ac-count ad-min	Run this command in the btp CLI: <code>btp create accounts/subaccount --display-name &lt;my-subaccount&gt; --region &lt;my-region&gt; --subdomain &lt;my-subaccount-subdomain&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> and <a href="#">Account Model</a> .
6.	View the details of the subaccounts in your global account.	Global ac-count ad-min or viewer	Run this command in the btp CLI: <code>btp get accounts/subaccount &lt;ID of new subaccount&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> .
7.	Add admins to your subaccounts.	Subac-count ad-min	Assign the role collection <code>Subaccount Administrator</code> to the user by running the following command in the btp CLI:  <code>btp assign security/role-collection "Subaccount Administrator" --to-user &lt;user&gt; --create-user-if-missing</code>	See <a href="#">Managing Users and Their Authorizations Using the btp CLI [page 2366]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 2202]</a> .
8.	View all the services and applications that are entitled to your global account, including quota information per service plan.	Global ac-count ad-min or viewer	Run this command in the btp CLI: <code>btp list accounts/entitlement</code>	See <a href="#">Setting Entitlements Using the btp CLI [page 2359]</a> .

Step No.	Task	Per-formed By	Do This	More Information
9.	Assign quotas to your subaccounts.	Global ac-count ad-min	Run this command in the btp CLI: <pre>btp assign accounts/entitlement --to- subaccount &lt;my-subaccount- id&gt; --for-service &lt;my- service&gt; --plan &lt;my- service-plan&gt; --amount &lt;number&gt;</pre> Validate with this command:	See <a href="#">Setting Entitle-ments Using the btp CLI [page 2359]</a> .
10.	View all the entitlements in your subaccounts.	Subac-count ad-min or viewer	Run this command in the btp CLI: <pre>btp list accounts/ entitlement --subaccount &lt;my-subaccount-id&gt;</pre>	See <a href="#">Setting Entitle-ments Using the btp CLI [page 2359]</a> .
11.	Create a Cloud Foundry org (environment instance) in your subaccounts.	Subac-count ad-min	Run this command in the btp CLI: <pre>btp create accounts/ environment-instance -- subaccount &lt;my-subaccount- id&gt; --display-name &lt;my- environment&gt; --environment &lt;cloudfoundry&gt;</pre>	See <a href="#">Working with En-vironments Using the btp CLI [page 2360]</a> .
12.	View details of the environment instances in your subaccounts.	Subac-count ad-min or viewer	Run this command in the btp CLI: <pre>btp list accounts/ environment-instance -- subaccount &lt;my-subaccount- id&gt;</pre>	See <a href="#">Working with En-vironments Using the btp CLI [page 2360]</a> .
13.	Create a Cloud Foundry space.	Org man-ager	Run these cf CLI commands: <pre>cf login cf create-space</pre>	See <a href="#">Create Spaces Using the Cloud Foun-dry Command Line In-terface [page 2485]</a> .

Step No.	Task	Performed By	Do This	More Information
14.	Add Cloud Foundry org and space members.	Org manager	Run these cf CLI commands: <pre>cf set-org-role &lt;USERNAME&gt;&lt;ORG&gt; &lt;ROLE&gt; cf set-space-role &lt;USERNAME&gt;&lt;ORG&gt;&lt;SPACE&gt;&lt;ROLE&gt;</pre>	See <a href="#">Add Organization Members Using the Cloud Foundry Command Line Interface [page 2483]</a> and <a href="#">Add Space Members Using the Cloud Foundry Command Line Interface [page 2486]</a> .
15.	Display all available services in the Cloud Foundry marketplace.	Org manager	Run this cf CLI command: <pre>cf marketplace</pre>	See <a href="#">Using Services in the Cloud Foundry Environment [page 248]</a> .

Using the btp CLI, you can perform account maintenance tasks, such as updating global account and subaccount details, deleting subaccounts, and deleting environment instances.

Subaccount members can also use the btp CLI to work with multitenant applications. See [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#).

## Next Steps

Org/space members can create service instances, which are entitled to the subaccount, using also the cf create-service <allowed-service-plan> command in the cf CLI. Use the cf CLI command cf services to verify that the service instances exist.

For more information about developer tasks, see [Development in the Cloud Foundry Environment \[page 233\]](#).

## Related Information

[Command Syntax of the btp CLI \[page 2329\]](#)

[Download and Start Using the btp CLI Client \[page 2323\]](#)

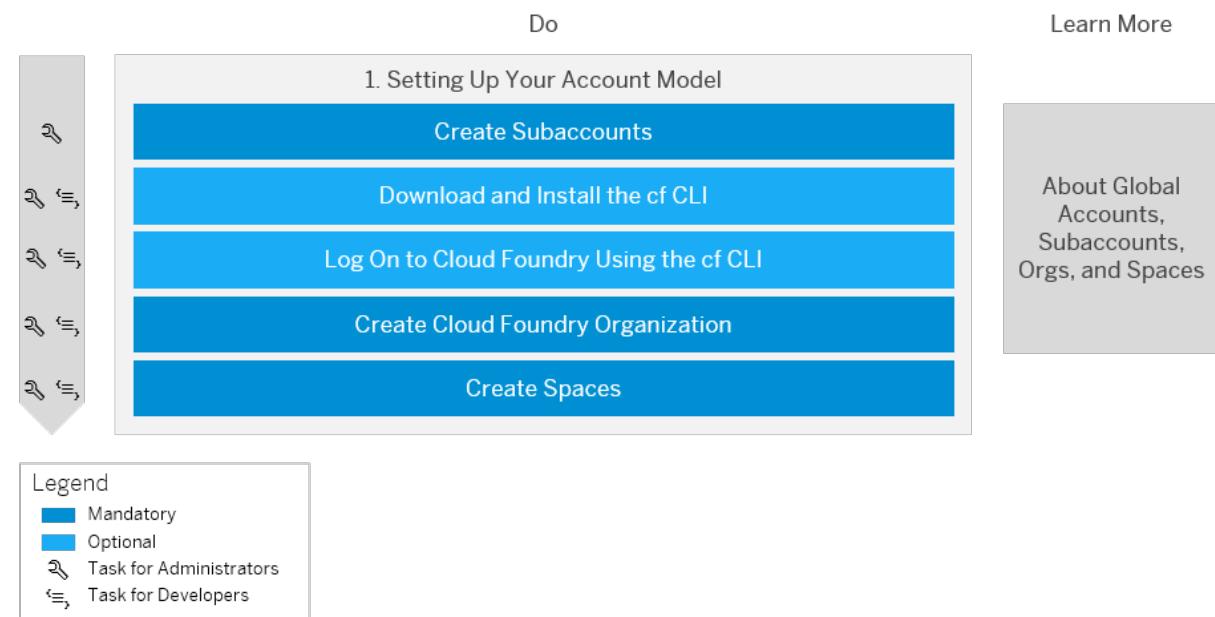
[Commands in the btp CLI \[page 2354\]](#)

## 3.2.2 Getting Started with an Enterprise Account in the Cloud Foundry Environment

Quickly get started with an enterprise account in the Cloud Foundry Environment.

This topic focuses on how to get started with a customer or partner account using the SAP BTP cockpit. However, you can also perform these tasks using the CLI. See [Setting Up a Global Account via the Command Line \[page 162\]](#).

### 1. Setting Up Your Account Model

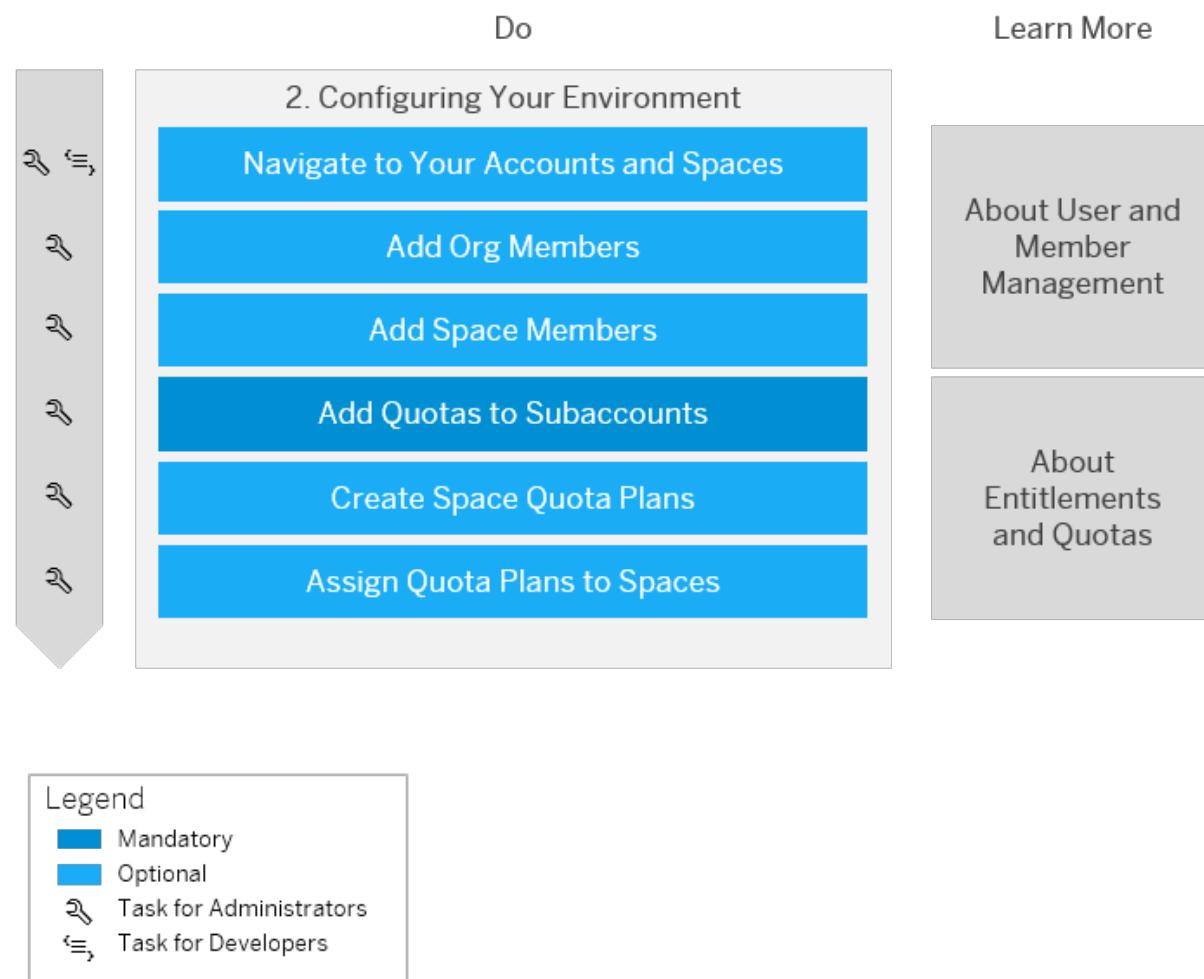


- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
  - [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)
  - [Create Spaces \[page 2441\]](#)
  - [Global Accounts \[page 94\]](#)
  - [Create a Subaccount \[page 2173\]](#)
  - [Create Orgs \[page 2435\]](#)
1. After you've received your logon data by email, create subaccounts in your global account. This allows you to further break down your account model and structure it according to your business needs. See [Create a Subaccount \[page 2173\]](#).
  2. If you haven't done so already, now is a good time to download and install the Cloud Foundry Command Line Interface (cf CLI). This tool allows you to administer and configure your environment, enable services, and deploy applications. See [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).

But don't worry, you can also perform all the necessary task using the SAP BTP cockpit, which you don't need to install.

3. If you'd like to use the cf CLI, log on to your environment. See [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
4. Create a Cloud Foundry organization in each of your subaccounts. See [Create Orgs \[page 2435\]](#)
5. Create spaces. See [Create Spaces \[page 2441\]](#).

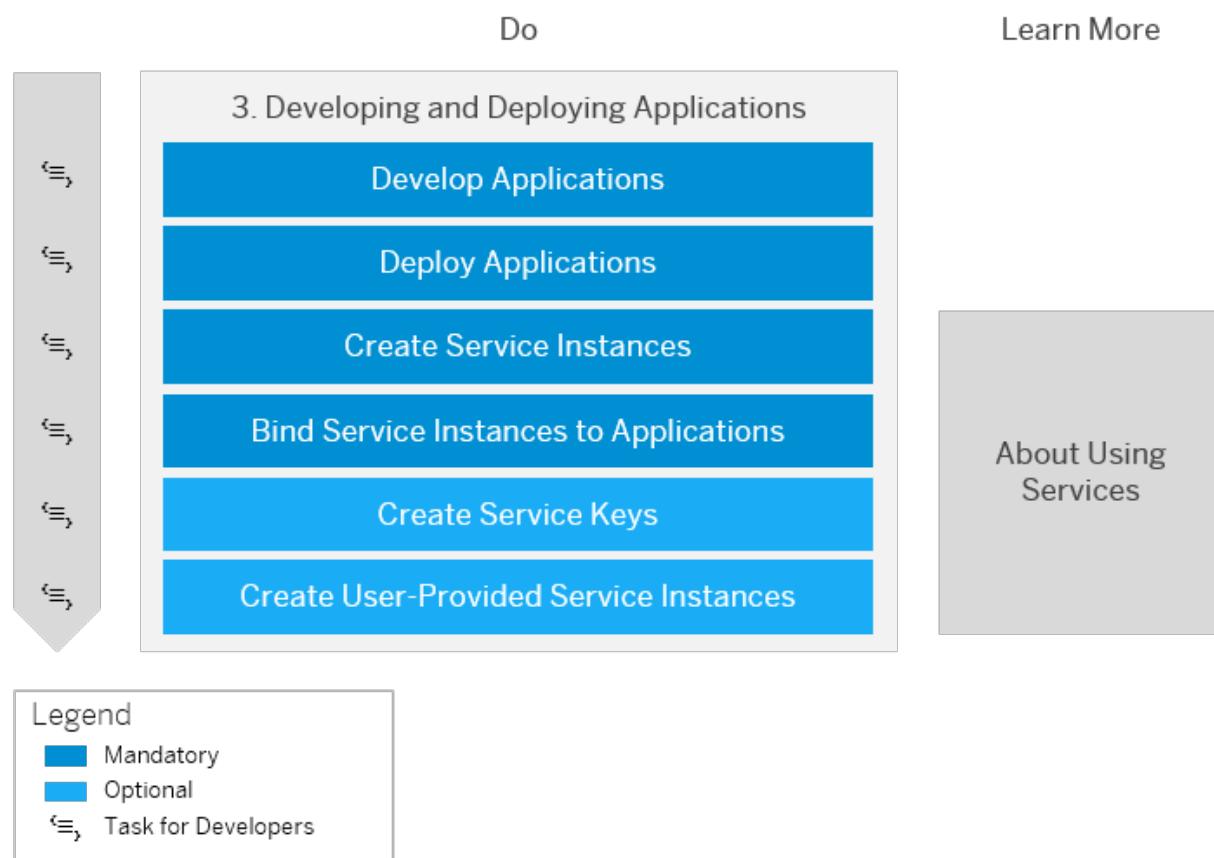
## 2. Configuring Your Environment



- [User and Member Management \[page 104\]](#)
- [Navigate to Orgs and Spaces \[page 2433\]](#)
- [Add Org Members \[page 2437\]](#)
- [Add Space Members \[page 2444\]](#)

- [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)
  - [Create Space Quotas \[page 2449\]](#)
  - [Assign Space Quotas to Spaces \[page 2450\]](#)
  - [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)
1. You can either use the cockpit or the cf CLI to configure your environment. If you'd like to use the cockpit, it's important you understand how you can navigate to your accounts and spaces. See [Navigate to Orgs and Spaces \[page 2433\]](#).
  2. It's time to think about member management. You can add members at different levels. For example, you can add members at an org level. See [Add Org Members \[page 2437\]](#). For more information about roles, see [User and Member Management \[page 104\]](#).
  3. You can also add members at a space level. See [Add Space Members \[page 2444\]](#).
  4. Before you can start using resources such as services or application runtimes, you need to manage your entitlements and add quotas to your subaccounts. See [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#). To learn more about entitlements and quotas, see [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#). Note that if you want to try out services for free, you need to select free tier service plans if available. For a list of free services, check the [SAP Discovery Center](#).
  5. You can also assign quotas to different spaces within a subaccount. To do so, first create a space quota plan. See [Create Space Quotas \[page 2449\]](#) or [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#).
  6. Then assign the quota plan to your space. See [Assign Space Quotas to Spaces \[page 2450\]](#) or [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 2488\]](#).

### 3. Developing and Deploying Applications



- [Creating Service Instances \[page 250\]](#)
  - [Binding Service Instances to Applications \[page 253\]](#)
  - [Creating Service Keys \[page 255\]](#)
  - [Using Services in the Cloud Foundry Environment \[page 248\]](#)
  - [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#)
  - [Development \[page 231\]](#)
  - [Creating User-Provided Service Instances \[page 252\]](#)
1. Develop your application. Check out the Developer Guide for tutorials and more information. See [Development \[page 231\]](#).
  2. Deploy your application. See [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#).
  3. Integrate your application with a service. To do so, first create a service instance. See [Creating Service Instances \[page 250\]](#)
  4. Bind the service instance to your application. See [Binding Service Instances to Applications \[page 253\]](#).
  5. Alternatively, you can also create and use service keys. See [Creating Service Keys \[page 255\]](#). For more information on using services and creating service keys, see [About Services \[page 249\]](#).

6. You can also create instances of user-provided services. See [Creating User-Provided Service Instances \[page 252\]](#).

### 3.2.2.1 Setting Up a Global Account via the Command Line

Your global account is the entry point for managing the resources, landscape, and entitlements for your departments and projects in a self-service manner in SAP BTP. You can use the command-line tool **btp CLI** to set it up, and the **cf CLI** to manage Cloud Foundry instances.

Set up your account model using the btp CLI by creating subaccounts in your enterprise account. You can create any number of subaccounts in the Cloud Foundry environment and region.

To manage Cloud Foundry, i.e. for managing service instances and members in orgs and spaces, creating spaces, as well as assigning quota to orgs and spaces, you use the cf CLI.

#### Prerequisites

- You have purchased an enterprise global account.
- You have downloaded and extracted the following command-line tools:
  - SAP BTP command line interface (btp CLI). See [Download and Start Using the btp CLI Client \[page 2323\]](#)
  - Cloud Foundry CLI (cf CLI). See [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

#### Procedure

##### → Tip

In the btp CLI, you can view the command help of each command to get information about how to use the command, its syntax, and input parameters. See [Get Help \[page 2336\]](#).

Step No.	Task	Performed By	Do This	More Information
1.	Log in to your global account using the URL of the btp CLI server and the subdomain of your global account.	Global account administrator or viewer	Run this command in the btp CLI: <code>btp login</code>	See <a href="#">Log in [page 2337]</a> .

Step No.	Task	Per-formed By	Do This	More Information
2.	View details of your global account.	Global ac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp get accounts/global-account</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> .
3.	Add admins to your global account.	Global ac-count ad-min	Assign the role collection <code>Global Account Administrator</code> to a user by running the following command in the btp CLI:  <code>btp assign security/role-collection "Global Account Administrator" --to-user &lt;user&gt; --create-user-if-missing</code>	See <a href="#">Managing Users and Their Authorizations Using the btp CLI [page 2366]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 2202]</a> .
4.	View all the regions that are available to your global account and subaccounts.	Global ac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp list accounts/available-region</code>	This command also provides information about the environments and infrastructure provider of each region.
5.	Create subaccounts in your global account.	Global ac-count ad-min	Run this command in the btp CLI:  <code>btp create accounts/subaccount --display-name &lt;my-subaccount&gt; --region &lt;my-region&gt; --subdomain &lt;my-subaccount-subdomain&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> .
6.	View the details of the subaccounts in your global account.	Global ac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp get accounts/subaccount &lt;ID of new subaccount&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the btp CLI [page 2355]</a> .
7.	Add admins to your subaccounts.	Subac-count ad-min	Assign the role collection <code>Subaccount Administrator</code> to the user by running the following command in the btp CLI:  <code>btp assign security/role-collection "Subaccount Administrator" --to-user &lt;user&gt; --create-user-if-missing</code>	See <a href="#">Managing Users and Their Authorizations Using the btp CLI [page 2366]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 2202]</a> .

Step No.	Task	Per-formed By	Do This	More Information
8.	View all the services and applications that are entitled to your global account, including quota information per service plan.	Global ac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp list accounts/ entitlement</code>	See <a href="#">Setting Entitlements Using the btp CLI [page 2359]</a> .
9.	Assign quotas to your subaccounts.	Global ac-count ad-min	Run this command in the btp CLI:  <code>btp assign accounts/entitlement --to- subaccount &lt;my-subaccount- id&gt; --for-service &lt;my- service&gt; --plan &lt;my- service-plan&gt; --amount &lt;number&gt;</code>  Validate with this command:	See <a href="#">Setting Entitlements Using the btp CLI [page 2359]</a> .
10.	View all the entitlements in your subaccounts.	Subac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp list accounts/ entitlement --subaccount &lt;my-subaccount-id&gt;</code>	See <a href="#">Setting Entitlements Using the btp CLI [page 2359]</a> .
11.	Create a Cloud Foundry org (environment instance) in your subaccounts.	Subac-count ad-min	Run this command in the btp CLI:  <code>btp create accounts/ environment-instance -- subaccount &lt;my-subaccount- id&gt; --display-name &lt;my- environment&gt; --environment &lt;cloudfoundry&gt; --plan standard --parameters " {\\"instance_name\\": \\"myOr g\\\" } "</code>	See <a href="#">Working with Environments Using the btp CLI [page 2360]</a> and <a href="#">Org Management Using the SAP BTP Command Line Interface (btp CLI) [page 2423]</a> .
12.	View details of the environment instances in your subaccounts.	Subac-count ad-min or viewer	Run this command in the btp CLI:  <code>btp list accounts/ environment-instance -- subaccount &lt;my-subaccount- id&gt;</code>	See <a href="#">Working with Environments Using the btp CLI [page 2360]</a> .
13.	Create a Cloud Foundry space.	Org manager	Run these cf CLI commands:  <code>cf login cf create-space</code>	See <a href="#">Create Spaces Using the Cloud Foundry Command Line Interface [page 2485]</a> .

Step No.	Task	Performed By	Do This	More Information
14.	Add Cloud Foundry org and space members.	Org manager	Run these cf CLI commands: <pre>cf set-org-role &lt;USERNAME&gt;&lt;ORG&gt; &lt;ROLE&gt; cf set-space-role &lt;USERNAME&gt;&lt;ORG&gt;&lt;SPACE&gt;&lt;ROLE&gt;</pre>	See <a href="#">Add Organization Members Using the Cloud Foundry Command Line Interface [page 2483]</a> and <a href="#">Add Space Members Using the Cloud Foundry Command Line Interface [page 2486]</a> .
15.	Display all available services in the Cloud Foundry marketplace.	Org manager	Run this cf CLI command: <pre>cf marketplace</pre>	See <a href="#">Using Services in the Cloud Foundry Environment [page 248]</a> .

Using the btp CLI, you can perform additional account maintenance tasks, such as updating global account and subaccount details, deleting subaccounts, and deleting environment instances.

Subaccount members can also use the btp CLI to work with multitenant applications. See [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#).

## Next Steps

Org/space members can create service instances, which are entitled to the subaccount, using also the cf create-service <allowed-service-plan> command in the cf CLI. Use the cf CLI command cf services to verify that the service instances exist.

For further documentation about developer tasks, see [Development in the Cloud Foundry Environment \[page 233\]](#).

## Related Information

[Command Syntax of the btp CLI \[page 2329\]](#)

[Download and Start Using the btp CLI Client \[page 2323\]](#)

[btp CLI Command Reference](#)

## 3.3 Getting Started in the ABAP Environment

Get onboarded in the SAP BTP, ABAP environment. Follow the workflows for trial or customer accounts.

[Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#)

After you have purchased a customer account, learn how to get started in the ABAP environment.

#### [Getting Started with Custom Code Analysis in the ABAP Environment \[page 222\]](#)

Learn how to get started with Custom Code Analysis in the SAP BTP, ABAP environment.

### **3.3.1 Getting Started with a Customer Account in the ABAP Environment**

After you have purchased a customer account, learn how to get started in the ABAP environment.

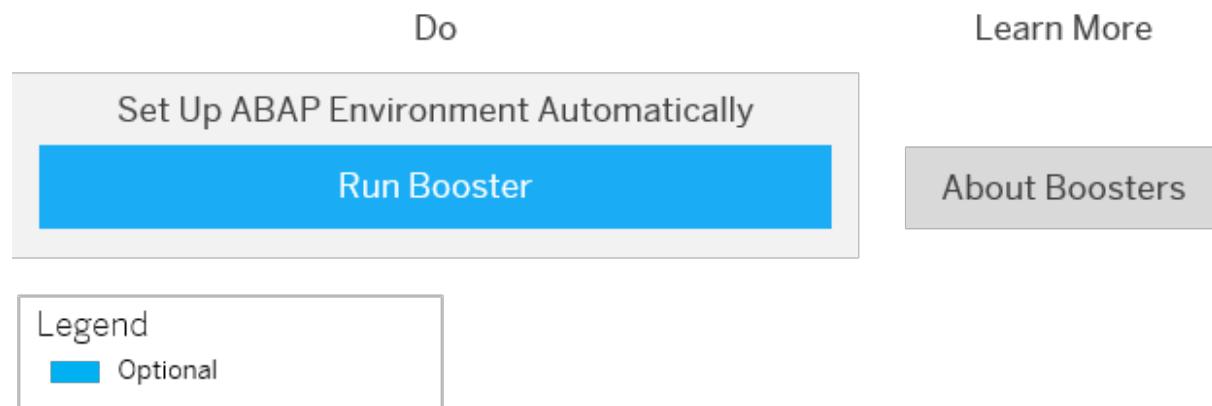
#### **ⓘ Note**

This documentation informs you about the first steps as an administrator in the ABAP environment. For more information about getting started as a developer, see [Getting Started as a Developer in the ABAP Environment \[page 217\]](#).

#### **1. Logging on to SAP BTP**

1. After you have purchased your customer account, you will receive an e-mail with a link to the home page of SAP BTP and the logon data as administrator for the global account.
2. Log on to SAP BTP.  
You are now in the SAP BTP cockpit.
3. Navigate to your global account.

#### **2. Automated Initial Setup Using a Booster (Optional)**



- Using a Booster to Automate the Setup of the ABAP Environment (Optional) [page 173]
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/fb1b56148f834749a2bf51127421610b.html> [<https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/fb1b56148f834749a2bf51127421610b.html>]

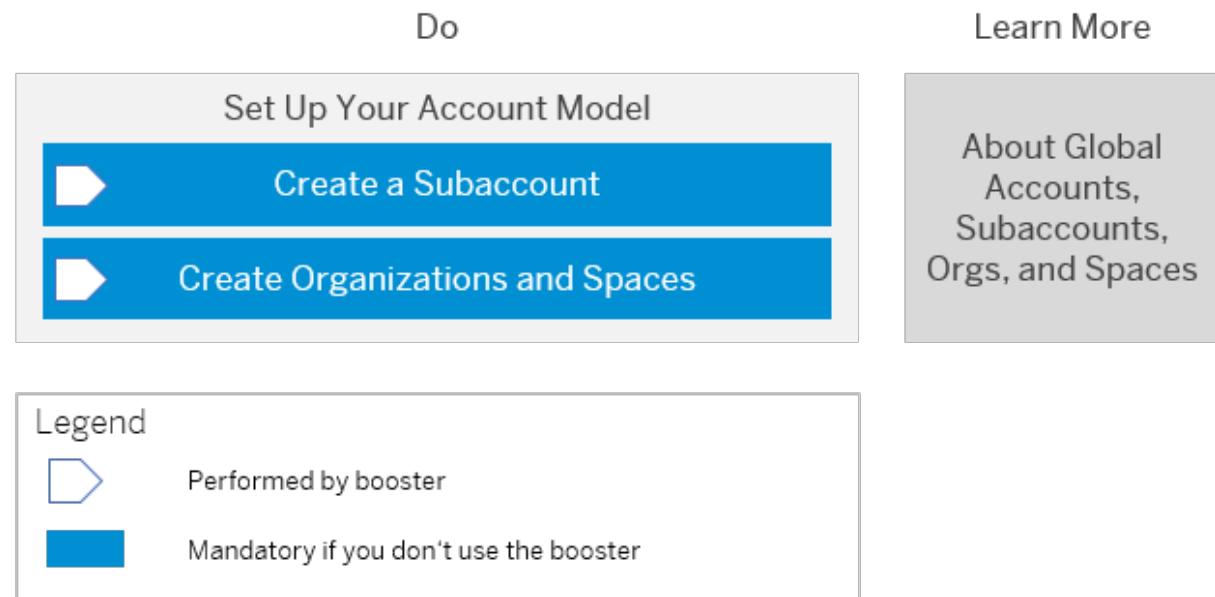
To set up a development system quickly, you can use the booster *Prepare an Account for ABAP Development*. The interactive booster guides you through the process of setting up your subaccounts, configuring entitlements, assigning members, and so on. For more information, see [Using a Booster to Automate the Setup of the ABAP Environment \(Optional\) \[page 173\]](#).

Using a booster is optional. If you don't use the booster, you must perform more steps manually.

### 3. Manual Setup

Even if you have used the booster, some manual setup steps are still required. In the following, steps that are performed by the booster are indicated.

#### a. Setting Up Your Account Model



- Creating a Cloud Foundry Subaccount for the ABAP Environment [page 175]
  - Creating a Cloud Foundry Organization and Space [page 176]
  - <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/8ed4a705efa0431b910056c0acdbf377.html> [<https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/8ed4a705efa0431b910056c0acdbf377.html>]
1. (Not required if you have used the booster):
 

Create a subaccount for ABAP environment in your global account (see [Creating a Cloud Foundry Subaccount for the ABAP Environment \[page 175\]](#) ).

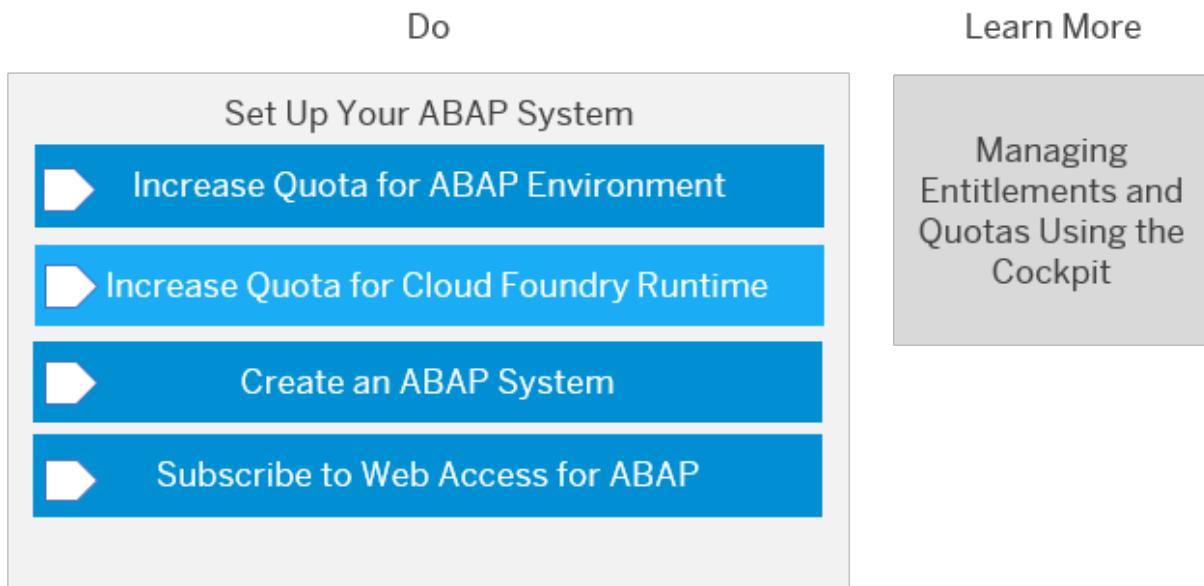
This allows you to further break down your account model and structure it according to your business needs.

2. (Not required if you have used the booster):

Create organizations and spaces (see [Creating a Cloud Foundry Organization and Space \[page 176\]](#)).

If you want to learn more about subaccounts, orgs, and spaces, and how they relate to each other, see [Account Model](#).

## b. Setting Up the ABAP System



### Legend



Performed by booster



Mandatory if you don't use the booster



Optional if you don't use the booster

- [Increasing the Quota for the ABAP Environment \[page 177\]](#)
- [Increasing the Quota for the Cloud Foundry Runtime \(Optional\) \[page 178\]](#)
- [Creating an ABAP System \[page 180\]](#)
- [Subscribing to the Web Access for ABAP \[page 186\]](#)
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/c8248745dde24afb91479361de336111.html> [<https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/c8248745dde24afb91479361de336111.html>]

1. (Not required if you have used the booster):

Before you can start using resources such as services or application runtimes, you must manage your entitlements and add quotas to your subaccount (see [Increasing the Quota for the ABAP Environment \[page 177\]](#) ).

2. (Not required if you have used the booster):

Optionally, if your developers want to deploy their own apps in Cloud Foundry, increase the quota for the Cloud Foundry runtime (see [Increasing the Quota for the Cloud Foundry Runtime \(Optional\) \[page 178\]](#) ).

3. (Not required if you have used the booster):

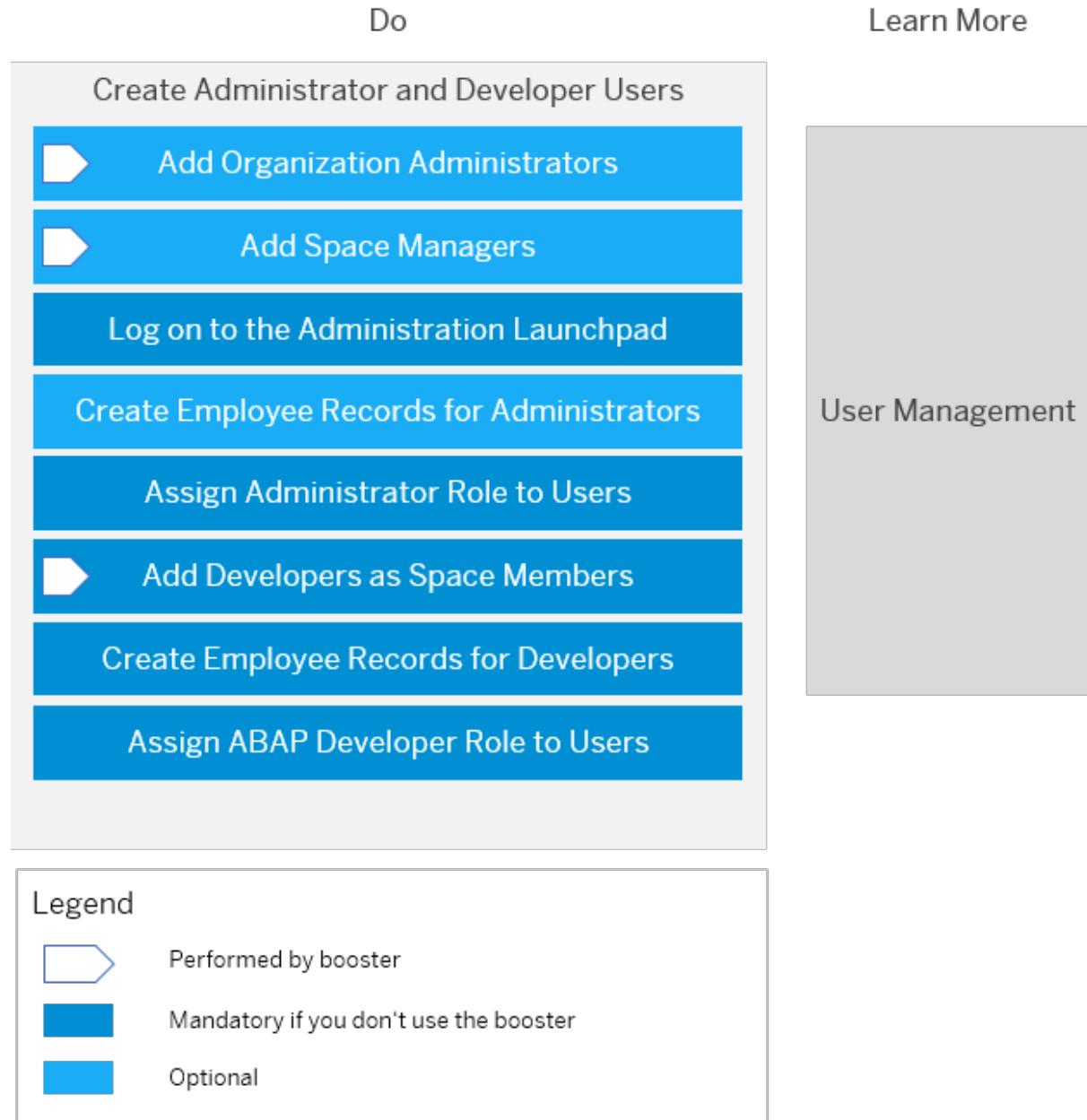
Create your ABAP system (see [Creating an ABAP System \[page 180\]](#) ).

4. (Not required if you have used the booster):

Subscribe to the Web access for ABAP SaaS application to get direct browser access to your instances in the ABAP environment (see [Subscribing to the Web Access for ABAP \[page 186\]](#) ).

This also allows you to access the administration launchpad including your own SAP Fiori applications. You only have to subscribe once for each subaccount.

### c. Creating Administrator and Developer Users



- Adding a User as Org Manager for the Cloud Foundry Organization [page 187]
- Adding a User as Space Manager for the Cloud Foundry Space [page 188]
- Logging on to the Administration Launchpad of the ABAP Environment [page 188]
- Creating an Employee Record for a New Administrator [page 189]
- Assigning the ABAP Environment Administrator Role to the New Administrator User [page 190]
- Creating New Space Members and Assigning Space Developer Roles to Them [page 191]
- Creating an Employee Record for a New Developer [page 191]

- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/cc1c676b4390406abb2a4838cbd0c37.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/cc1c676b4390406abb2a4838cbd0c37.html]
  - [Assigning the ABAP Developer User to the ABAP Developer Role \[page 192\]](#)
1. Optionally, you can create additional administrator users for the ABAP environment as follows:
    1. (Not required if you have used the booster):  
Add users as administrators of the Cloud Foundry organization that you created for the ABAP environment (see [Adding a User as Org Manager for the Cloud Foundry Organization \[page 187\]](#)).
    2. (Not required if you have used the booster):  
Add the users as managers for the Cloud Foundry space that you created for the ABAP environment (see [Adding a User as Space Manager for the Cloud Foundry Space \[page 188\]](#) ).
    3. Log on to the administration launchpad (see [Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)).
    4. Create employee records for the new administrators (see [Creating an Employee Record for a New Administrator \[page 189\]](#)).
    5. Assign the administrator role for the ABAP environment to the new users (see [Assigning the ABAP Environment Administrator Role to the New Administrator User \[page 190\]](#)).
  2. For your developers, create business users and assign them to the developer business role as follows:
    1. (Not required if you have used the booster):  
Create new members for the space that you created for the ABAP environment and assign developer roles to the new members (see [Creating New Space Members and Assigning Space Developer Roles to Them \[page 191\]](#)).
    2. Create employee records for the new developer users (see [Creating an Employee Record for a New Developer \[page 191\]](#)).
    3. Assign the ABAP developer users to the ABAP developer role (see [Assigning the ABAP Developer User to the ABAP Developer Role \[page 192\]](#)).

## 4. What's Next

After the initial setup of the ABAP environment, your developers can now develop applications using ABAP Development Tools. You can tell your developers to get started (see [Getting Started as a Developer in the ABAP Environment \[page 217\]](#)).

As next steps, you can optionally set up additional tools for your developers or ensure the integration with other systems:

- If your developers need a UI development tool, set up SAP Business Application Studio (see [Setup of UI Development in SAP Business Application Studio \(Optional\) \[page 194\]](#)).
- As a default identity service in the ABAP environment, you get SAP ID. You might want to set up a custom identity service (see [Setup of a Custom Identity Service \[page 202\]](#)).
- As an alternative to SAP Business Application Studio, you can also set up SAP Web IDE (see [Setup of UI Development in SAP Web IDE \(Optional\) \[page 208\]](#)).

Setup of UI  
Development in SAP  
Business Application  
Studio

Setup of a Custom  
Identity Provider

Setup of UI  
Development in SAP  
Web IDE

- Setup of UI Development in SAP Business Application Studio (Optional) [page 194]
- Setup of a Custom Identity Service [page 202]
- Setup of UI Development in SAP Web IDE (Optional) [page 208]

## Related Information

[Learning Journey](#)

[Discovery Center](#) 

[SAP Community](#) 

[SAP Road Map Explorer](#) 

[Video Tutorials](#) 

### 3.3.1.1 Using a Booster to Automate the Setup of the ABAP Environment (Optional)

You can use a booster to automate some of the required steps for setting up the ABAP environment. Automation includes creating a subaccount and space, configuring the required entitlements, and assigning administrators and developers to the subaccount.

#### Context

A booster is a set of guided interactive steps that enable you to select, configure, and consume services on SAP BTP. For more information, see [Boosters](#).

##### ⚠ Caution

The booster is only intended to create development systems. For non-development systems, follow the steps to set up the ABAP environment manually (see [Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#)).

#### Procedure

1. Log on to the SAP BTP cockpit and choose the global account for the Cloud Foundry environment as administrator.
  2. From the navigation menu, choose [Boosters](#).
  3. Choose the booster [Prepare an Account for ABAP Development](#).
- The tab pages [Overview](#), [Components](#), and [Additional Resources](#) are displayed, where you get more information about the booster.
4. Choose [Start](#).
  5. After the booster wizard has successfully checked the prerequisites, choose [Next](#) to continue.
  6. On the [Select Scenario](#) dialog, choose whether you'd like the booster to create your ABAP environment service instance in a new, or in an existing subaccount.
  7. On the [Configure Subaccount](#) dialog, entitlements and quotas for the new subaccount are proposed, but you can change them if needed:
    - For the [ABAP environment](#) service, the service plan [standard](#) enables you to size ABAP server and persistence independently from each other in 16 GB units. These units are represented in the quota plans [abap\\_compute\\_unit](#) and [hana\\_compute\\_unit](#).  
As part of the default quota assignment, you get at least 1 compute unit in the [abap\\_compute\\_unit](#) service plan and at least 2 compute units in the [hana\\_compute\\_unit](#) service plan. This corresponds to the minimum configuration for an instance of the ABAP Environment service. You can also choose higher quotas in the dialog of the booster.
    - For the [Web Access to ABAP](#) service, a default plan is chosen that you cannot change. You need the service and the quota for direct browser access to your instances in the ABAP environment, including access to the administration launchpad for ABAP.

- Depending on what you have ordered for your account, additional entitlements for the services *Cloud Foundry Runtime*, **SAP Build Work Zone, standard edition**, and *SAP Business Application Studio* might be shown. They are optional; you can remove the entitlements if you don't need them right now and want to add them later:
    - A quota for the Cloud Foundry runtime is only needed for the ABAP environment if your developers want to deploy their own apps in Cloud Foundry.
    - The SAP Build Work Zone, standard edition enables organizations to establish a central point of access to SAP, custom-built, and third party applications and extensions, both in the cloud and on-premise.
    - SAP Business Application Studio is a UI development environment.
- On the *Configure Subaccount* dialog, follow the instructions on the screen to enter subaccount name, provider, region, and so on.
  - On the *Add Users* dialog, add the e-mail addresses of new users in the ABAP environment.
- This step allows you to quickly create additional administration users for the ABAP environment, if needed, and developer users. In the *Origin* field, you can see the identity provider for the ABAP environment. By default, this is SAP ID service (*sap.ids*), but if you have set up a custom identity provider, you can choose this provider from the dropdown list.
- On the *Configure ABAP Environment Instance* dialog, enter a 3-character ID for your ABAP system.
  - Review your settings and finish the booster.

## Results

After the booster has run successfully, the following tasks have been performed automatically:

- A new Cloud Foundry subaccount for the ABAP environment is created and enabled.
- A space and an organization for the ABAP environment are available.
- A service instance for the ABAP environment has been created.
- A system ID of your choice has been assigned to the ABAP system.
- An identity provider (SAP ID service or a custom identity provider) has been set up for the ABAP environment.
- For the *ABAP environment* service, quotas have been distributed for the ABAP server and the SAP HANA persistence. The quotas that you have chosen are deducted from the available quota for each service in your global account..
- If you have entitlements for SAP Build Work Zone, standard edition and SAP Business Application Studio, then you have subscribed to these services and standard quotas have been assigned accordingly. Entitlements for subscription services are assigned to the subaccount.
- You're subscribed to the Web access for ABAP SaaS application and get direct browser access to your instances in the ABAP environment. Entitlements for subscription services are assigned to the subaccount.
- Additional users with Cloud Foundry organization manager and space manager roles have been added. If you have ordered the SAP Build Work Zone, standard edition as a service, the additional administration users have also been assigned to the role collection *Launchpad\_Admin*.
- Additional users for developers with space developer role have been added. If you have ordered SAP Business Application Studio as a service, the additional developer users have also been assigned to the role collection *Business\_Application\_Studio\_Developer*.

#### Note

The booster speeds up the initial setup, but there are still some tasks that you need to perform after the booster has run.

In the following, you can find documentation for all necessary intial tasks, including the tasks that you don't need to perform if you have run the booster successfully. A note at the beginning of each task indicates that you can skip it if you've used the booster.

### 3.3.1.2 ABAP Environment: Initial Settings

With the initial settings, you as an administrator and your ABAP developers can get started in the ABAP environment.

After the initial settings are done, the ABAP environment is ready for your ABAP developers so they can implement business services using ABAP Development Tools (ADT).

#### 3.3.1.2.1 Creating a Cloud Foundry Subaccount for the ABAP Environment

Create a Cloud Foundry subaccount for the ABAP environment in your global account using the cockpit.

#### Context

#### Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip these steps.

For the ABAP environment, you need a Cloud Foundry subaccount as a technical environment. If you want to reuse an existing Cloud Foundry subaccount, make sure that it uses the providers and regions that are available for provisioning an ABAP environment. For more information about the providers and regions for ABAP environment, see [Regions](#).

#### Procedure

1. From your global account, choose [New Subaccount](#).
2. Specify a display name, for example, **ABAP**.
3. **Optional:** Enter a description.
4. Leave the [Neo Environment](#) checkbox deselected.

5. Choose the desired infrastructure provider and region for your subaccount.

You can find the available providers and regions for the ABAP environment in [Regions](#).

6. Enter a subdomain for your subaccount. The subdomain will become part of the URL for accessing applications that you subscribe to from this subaccount.

#### Note

You can choose any string of your choice. However, note that the subdomain can contain only letters, digits and hyphens (not allowed in the beginning or at the end), and must be unique across all subaccounts in the same region. Uppercase and lowercase letters can be used, however that alone does not qualify as a means to differentiate subdomains (for example, **SUBDOMAIN** and **subdomain** are considered to be the same).

7. If your subaccount is to be used for production purposes, select the [Used for production](#) option.
8. Choose [Create](#).

### 3.3.1.2.2 Creating a Cloud Foundry Organization and Space

Create a Cloud Foundry organization and Cloud Foundry space that will correspond to the instance of the ABAP environment.

#### Context

##### Note

If you have already created an organization and space for the ABAP environment, you can skip these steps. If you have run the booster [Prepare an Account for ABAP Development](#), you can also skip these steps.

For more information about creating Cloud Foundry spaces, see [Create Spaces](#).

#### Procedure

1. Log on to the SAP BTP cockpit in the global account for the Cloud Foundry environment as administrator.
2. Choose the tile of the Cloud Foundry subaccount.
3. Check whether Cloud Foundry is enabled: If it's enabled, the Cloud Foundry organization name appears as organization on the screen area *Cloud Foundry*.
4. If Cloud Foundry is not enabled yet, proceed as follows:
  - a. Choose [Enable Cloud Foundry](#).
  - b. On the [Create Cloud Foundry Organization](#) dialog, enter a name for the Cloud Foundry organization and choose [Create](#).
  - c. Choose the free or standard plan for the Cloud Foundry runtime according to your needs.

5. On the [Cloud Foundry](#) screen area, choose the link for spaces.
6. If the Cloud Foundry space in the Cloud Foundry organization does not exist yet, proceed as follows:
  - a. Choose [New Space](#).
  - b. On the [New Space](#) dialog, enter a name for the new Cloud Foundry space, for example, `dev`, and leave the checkboxes for space manager and space developer checked.
  - c. Choose [Create](#).

### 3.3.1.2.3 Increasing the Quota for the ABAP Environment

Before you can create a service instance for the ABAP environment, you must assign some of the available quota to the subaccount for the ABAP environment.

#### Context

##### Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

If you are working in an enterprise account, you need to add quotas to the services that you purchased in your subaccount before they can appear in the service marketplace. For more information, see [Configure Entitlements and Quotas for Subaccounts](#).

#### Procedure

1. Log on to the SAP BTP cockpit as administrator.
2. Go to your global account.
3. From the navigation area, choose  [Entitlements](#)  [Entity Assignments](#).
4. Enter the subaccount for the ABAP environment and choose [Go](#).
5. Choose [Configure Entitlements](#).
6. Choose [Add Service Plans](#).

In the following popup, the available entitlements for this subaccount are shown.

7. For the [ABAP environment](#) entitlement, select the plans `abap_compute_unit, standard`, and `hana_compute_unit`.

With the selection of the `standard` service plan, you can size the ABAP runtime and the SAP HANA memory independently from each other. To be able to do your sizing, you must also select the quota plans `abap_compute_unit` and `hana_compute_unit`.

8. Choose [Add 3 Service Plans](#).
9. On the following screen, increase the quotas in the `abap_compute_unit` service plan at least by 1 and in the `hana_compute_unit` service plan at least by 2.

The minimum configuration of 1 ABAP compute unit and 2 HANA compute units is sufficient in many cases to create one service instance for the ABAP environment. If you need more resources, choose higher quotas. Keep in mind that increasing the SAP HANA Cloud storage using the parameter `size_of_persistence_disk` consumes 0.002 HANA compute units for each GB that exceeds the minimal size of the persistence disk.

10. Choose [Save](#).

## Results

After you have assigned an entitlement for the three plans with respective quotas to your subaccount, you can create an ABAP system. This is the point where you can then decide how many blocks of ABAP compute units and HANA compute units you want to assign to your new system. See [Creating an ABAP System \[page 180\]](#).

## Related Information

[ABAP Compute Units \[page 2983\]](#)

### 3.3.1.2.4 Increasing the Quota for the Cloud Foundry Runtime (Optional)

Check if your subaccount has a quota for the Cloud Foundry runtime and increase it, if needed.

## Context

### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

A quota for the Cloud Foundry runtime is optional. It's only needed for the ABAP environment if your developers want to deploy their own apps in Cloud Foundry.

## Procedure

1. Log on to the SAP BTP cockpit as Cloud Foundry administrator.
2. Go to your global account.
3. From the navigation area, choose [Entitlements](#) [Entity Assignments](#)

4. If there is no entry for the Cloud Foundry runtime, choose [Configure Entitlements](#) and [Add Service Plans](#).
5. In the following popup, proceed as follows:
  - a. Choose [Cloud Foundry Runtime](#).
  - b. Under [Available Service Plans](#), select the checkbox **MEMORY**.
  - c. Choose [Add 1 Service Plan](#).
6. Choose **+** to add at least 1 to the subaccount.
7. Choose [Save](#).

### 3.3.1.2.5 Selecting a Service Plan for the Web Access for ABAP

Before you can subscribe to the Web Access for the ABAP environment, you must select the relevant service plan.

#### Context

##### Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

If you are working in an enterprise account, you need to add quotas to the services that you purchased in your subaccount before they can appear in the service marketplace.

#### Procedure

1. Log on to the SAP BTP cockpit as administrator.
2. Go to your global account.
3. From the navigation area, choose  [Entitlements](#)  [Entity Assignments](#).
4. Enter the subaccount for the ABAP environment and choose [Go](#).
5. Choose [Configure Entitlements](#).
6. Choose [Add Service Plans](#).

In the following popup, the available entitlements for this subaccount are shown.

7. For the [Web Access for ABAP](#) entitlement, select the plan [default \(Application\)](#).
8. Choose [Add 1 Service Plan](#).
9. Choose [Save](#).

### 3.3.1.2.6 Creating an ABAP System

Create a service instance for the ABAP environment from the Service Marketplace.

#### Prerequisites

You have increased the quota for the ABAP environment. See [Increasing the Quota for the ABAP Environment \[page 177\]](#).

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can also skip this step.

For more information about creating service instances, see [Create Service Instances Using the Cockpit](#).

#### Procedure

1. Log on to the SAP BTP cockpit and navigate to the Cloud Foundry subaccount. See [Navigate in the Cockpit](#).
2. From the navigation area, choose [Services](#) [Service Marketplace](#). You see a list of all services that are available to you.
3. Choose [ABAP environment](#).
4. Choose [Create](#).

A wizard opens that helps you create your instance.

##### → Tip

To see a list of all instances that have already been created in your subaccount, choose [Services](#) [Instances and Subscriptions](#) from the navigation area. Here, you can also create a new ABAP environment instance.

5. Select the standard service plan.
6. Choose [Cloud Foundry](#) as your runtime environment.
7. Select a space.
8. Enter a CLI-friendly instance name, and choose [Next](#).

The instance name identifies the service instance in the Cloud Foundry environment. Specify an instance name that is unique among all the service instances in a space and only contains alphanumeric characters (A-Z, a-z), periods, underscores, and hyphens.

- Provide additional instance parameters for the configuration by either using the default **form** or by uploading a **JSON file** from your computer or by specifying the parameters in JSON format. Here is a list of all parameters:

<b>Parameter</b> (technical name in parenthesis)	<b>Note</b>
<b>Admin Email Address</b> (admin_email)	The <b>admin email address</b> is used to create the initial user for the ABAP system automatically, including the assignment of the administrator role to this user. You can access the ABAP environment system only with this specified user. By default, the email address is used as subject name identifier.
<b>Admin Employee ID</b> admin_employee_id	The <b>admin employee ID</b> is an optional parameter used to provide the employee ID for the initial user. You can either set it directly by using this parameter during instance creation or after instance creation in the Maintain Employees Fiori application of the ABAP environment service instance. Once set, it cannot be changed anymore. The employee ID must only contain uppercase letters, digits, underscores, hyphens and periods, and must not be longer than 60 characters.
<b>ABAP System Description</b> (description)	The <b>ABAP system description</b> is optional.
<b>Development System</b> (is_development_allowed)	The <b>development system</b> checkbox is checked by default. By using this setting, you can control the changeability of development objects in the system. If you want to protect all your customer-related software components and ABAP namespaces against manual changes via ABAP development tools for Eclipse, uncheck the box. This setting is used for test and productive systems, where changes must be imported only. For information about which business catalogs are available in development systems only, see <a href="#">Business Catalogs for Development Tasks [page 2548]</a> .
<b>ABAP System ID</b> (sapSystemname)	The <b>ABAP system ID</b> must consist of exactly three alphanumeric characters. Only uppercase letters are allowed. The first character must be a letter (not a digit). The ID does not have to be technically unique.
<b>Total ABAP Runtime Size</b> (size_of_runtime)	<p>The <b>Total ABAP runtime size</b> refers to the runtime size of the ABAP environment service instance. This is the sum of the runtime size of all ABAP application servers of an ABAP environment service instance. The size is specified in number of ABAP compute units that should be used from the quota plan <i>abap_compute_unit</i>, with one ABAP compute unit representing 16 GB. The supported number of abap_compute_unit is 1, 2, 4, 6, 8, 16, 24, or 32. For more information, see <a href="#">ABAP Compute Units [page 2983]</a>.</p> <p>The total ABAP runtime size should be a whole-number multiple, ranging from 2 to 16 times the size of the ABAP runtime size per application server. This is because each ABAP environment instance contains between 2 and 16 application servers, all of the same size. For example, if the <i>application_server_size</i> is 2, the <i>size_of_runtime</i> should be at least 4 ACUs. Conversely, if the <i>application_server_size</i> is 0.5, the <i>size_of_runtime</i> shouldn't exceed 8 ACUs. Another example: if the <i>application_server_size</i> is 2, setting the <i>size_of_runtime</i> to</p>

Parameter (technical name in parenthesis)	Note
	5 won't work. This is because a total runtime size of 5 ACUs can't be split into three application servers with 2 ACUs each.
<b>ABAP Runtime Size per Application Server</b> <b>(application_server_size)</b>	<p>The <i>ABAP Runtime Size per Application Server</i> refers to the size of a single ABAP application server in an ABAP environment service instance. It's specified in ABAP compute units (ACUs), with one ACU representing 16 GB. This parameter doesn't consume additional <code>abap_compute_unit</code> quota. Instead, it determines the distribution of the quota defined via the total ABAP runtime size among the application servers. Possible values are 0.5, 2, or 'auto', which is the default.</p> <p>A higher value results in fewer, but larger, application servers. Conversely, a lower number results in more, but smaller, application servers for the same total ABAP runtime size. If 'auto' is chosen, then 0.5 ACU is assigned per application server if <code>size_of_runtime</code> is &lt; 4 for systems without elastic scaling, or <code>size_of_runtime</code> &lt; 8 for systems with elastic scaling. Otherwise, 2 ACUs are assigned per application server.</p> <p>Every ABAP environment instance has between 2 and 16 application servers of the same size each. Therefore, the possible value for the ABAP runtime size per application server depends on the total ABAP runtime size. For instance, <code>application_server_size</code> = 2 can only be set if the <code>size_of_runtime</code> is at least 4 ACUs. On the other hand, <code>application_server_size</code> = 0.5 can only be set if the <code>size_of_runtime</code> isn't higher than 8 ACUs.</p> <p>To decide what value to choose, consider the following:</p> <ul style="list-style-type: none"> <li>• Servers of size 0.5 ACUs allow more fine-granular scaling and are suitable for most use cases.</li> <li>• Larger servers allow slightly more user sessions per ACU. This might be more efficient in systems with very high ACU consumption. They're also a precondition for some exceptional use cases, such as ATC checks for very large ABAP programs.</li> </ul>
<b>Elastic Scaling of the ABAP Application Server</b> <b>(elastic)</b>	<p>The <b>elastic scaling of the ABAP application server</b> parameter defines whether or not to adapt the number of ABAP application servers dynamically, depending on the system load.</p> <p>By default, the checkbox is not checked (<code>elastic = false</code>), and the service instance will have a fixed number of ABAP application servers with a total ABAP runtime size as defined by the parameter <b>Total ABAP Runtime Size</b> (see above).</p> <p>If you enable elastic scaling by checking the checkbox (<code>elastic = true</code>), application servers will be automatically added to the service instance when the load increases and removed when the load decreases. More specifically, the number of application servers will scale between the</p>

<b>Parameter</b> (technical name in parenthesis)	<b>Note</b>
	<p>number of ACUs (ABAP compute units) needed for two application servers as a minimum (1 ACU if <code>application_server_size = 0.5</code> or 4 ACU if <code>application_server_size = 2</code>), and the number of ACUs configured in the <b>Total ABAP Runtime Size</b> parameter (see above) as a maximum.</p>
	<p>This feature considerably helps in cost-saving as charges are only levied based on the actually allocated ACUs, and not on the configured maximum. Still, to ensure that defined quotas are not exceeded, the maximum number of ACUs that is set in the <b>Total ABAP Runtime Size</b> parameter is allocated from the available ACU quota for the entire duration that the service instance is provisioned.</p>
	<p>The scaling typically occurs within a couple of minutes. Especially when a service instance is scaled down, there is a grace period of a few minutes to allow the completion of short-lived requests. Before enabling elastic scaling in production, we advise to test your typical workload patterns and monitor the embedded <a href="#">Health Monitoring</a> app to verify the suitability of this cost-saving measure. Maintaining a higher number of statically configured ACUs might be better for handling sudden large spikes in the system load.</p>
<b>HANA Cloud Memory Size</b> <code>(size_of_persistence)</code>	<p>The <b>HANA Cloud memory size</b> refers to the memory size of the SAP HANA Cloud database used by the ABAP environment service instance. The size is specified in number of HANA compute units that should be used from the quota plan <a href="#">hana_compute_unit</a>, with one HANA compute unit representing the suitable block size for the underlying SAP HANA Cloud instance (16 GB on Microsoft Azure and Google Cloud Platform, or 15 GB on AWS). The supported number of <code>hana_compute_unit</code> per HANA instance is 2, 4, 8, 16, 32, or 64.</p>
<b>HANA Cloud Disk Size</b> <code>(size_of_persistence_disk)</code>	<p>The <b>HANA Cloud disk size</b> refers to the disk size in GB of the SAP HANA Cloud database used by the ABAP environment service instance. If the parameter is set to <code>auto</code>, the SAP HANA Cloud storage size is set to the minimal value <code>40 * size_of_persistence + 40</code>. The maximum allowed value is <code>120 * size_of_persistence + 40</code>. If you set a higher value, it will consume 0.002 HANA compute units (HCU) for any GB exceeding the minimal default size of the persistence disk. Therefore, the HCU ratio of additional storage disk to RAM is 1 : 31.25 per GB on Microsoft Azure and on Google Cloud Platform (as 1 HCU = 16 GB) or 1 : 33.33 per GB on AWS (as 1 HCU = 15 GB).</p>
<b>Admin User Name</b> <code>(admin_user_name)</code>	<p>The <b>admin user name</b> is an optional parameter used to provide the user name for the initial user. It must be provided if the <b>login attribute</b> is set to <code>user_name</code> (see below).</p>
<b>Login Attribute</b> <code>(login_attribute)</code>	<p>Using an email address as subject name identifier might not be possible if the e-mail address is ambiguous across users, or if the trusted identity provider configured for authentication in the subaccount of the ABAP</p>

Parameter (technical name in parenthesis)	Note
	<p>environment instance is already configured with the subject name identifier <b>Login Name</b>. In this case, you can change the <b>login attribute</b> to <b>user_name</b>. In addition, provide the user name for the initial user in the <b>admin user name</b> (see above).</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"><p><b>⚠ Caution</b></p><p>Currently, it's only possible to change the login attribute <b>when creating a new ABAP system</b>. To change the settings <b>afterwards</b>, please create a service ticket.</p></div>

#### ⓘ Note

Make sure that you don't choose more compute units than you have assigned to your subaccount for the ABAP environment (see [Increasing the Quota for the ABAP Environment \[page 177\]](#)).

For more information about ABAP system sizing and Native Storage Extension for selected SAP HANA tables, see [ABAP System Sizing \[page 2971\]](#) and [Native Storage Extension for the ABAP Environment \[page 2975\]](#).

#### ⚠ Caution

Be aware that not all parameters can be changed if you update your ABAP system. For a list of updatable parameters, see [Updating an ABAP System \[page 184\]](#).

10. Choose **Next** to review and verify your instance details.

11. Choose **Create**.

The ABAP environment instance is being set up, which might take a while. Wait for an email that is sent when the setup is completed and the system up and running. The email is sent to the email address that you specified as admin email in the previous steps.

### 3.3.1.2.6.1 Updating an ABAP System

Learn how to update your ABAP environment instance.

#### Context

After you have created an ABAP environment instance, you can change the following properties:

- ABAP System Description
- Total ABAP Runtime Size (size\_of\_runtime)

- ABAP Runtime Size per Application Server (application\_server\_size)
- Elastic Scaling of the ABAP Application Server
- HANA Cloud Memory Size (size\_of\_persistence)
- HANA Cloud Disk Size (size\_of\_persistence\_disk)

### Note

The updates of these properties are done without any downtime, or - in case of updating the HANA memory or disk size - with near zero downtime (meaning that the database is not accessible for a few seconds only).

To **change the initial admin**, you have to use the [Maintain Employees](#) SAP Fiori app. See [Maintain Employees \[page 2624\]](#).

To **update your system from a free to a standard plan**, see [Update to a Paid Service Plan](#). Make sure you have enough hana\_compute\_units and abap\_compute\_units available as quota in your subaccount before updating to a paid plan. For more information, see [Increasing the Quota for the ABAP Environment \[page 177\]](#).

### Restriction

A downsizing of the HANA Cloud memory size is only possible down to the minimum amount of memory required for a stable performance of the HANA Cloud database. Before the memory size is decreased, an SQL query (as described on [SAP HANA Database Downsizing](#)) is automatically run on the database to make sure the downsizing is possible based on the already consumed memory.

The HANA Cloud memory size cannot be decreased while a system is stopped. To decrease the HANA Cloud memory size of a stopped instance, the instance has to be started via the [Landscape Portal](#) first.

The HANA Cloud disk size can currently only be increased, and not decreased.

Both the HANA Cloud memory and the disk size can only be resized once every 24 hours.

## Procedure

1. In the SAP BTP cockpit, choose your Cloud Foundry subaccount and navigate to the space, in which you have created your ABAP environment instance. See [Navigate in the Cockpit](#).
2. From the navigation area, choose  You see a list of all instances that have been created.
3. Select your *ABAP environment* instance and choose *Update* by clicking on  at the end of the row. If you'd like to check the values that are currently configured first, please select *View Parameters* instead.
4. In the *Update Instance* wizard, you can update the parameters listed above. See [Increasing the Quota for the ABAP Environment \[page 177\]](#). The currently configured values are prefilled in the *Update Wizard*, so you can keep the values that you don't want to change as they are.
5. Choose *Update Instance* to save your changes.

## Results

Your ABAP environment instance is being updated. This might take a while.

### 3.3.1.2.7 Subscribing to the Web Access for ABAP

You need to subscribe to the [Web access for ABAP](#) SaaS application to get direct browser access to your instances in the ABAP environment, including access to the administration launchpad for ABAP.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

This subscription is required to access the SAP Fiori launchpad for the administrator and to access developed SAP Fiori applications. Subscription is required only once per subaccount (for all systems to be created in all spaces of this subaccount). After subscribing to the application, you have access to the administration launchpad, where you can configure application roles and assign those roles to your users.

#### Procedure

1. Log on to the SAP BTP cockpit in the Cloud Foundry subaccount of the global account for Cloud Foundry as administrator.
2. In the navigation area, choose [Services](#) [Service Marketplace](#).
3. Choose the SaaS application tile [Web access for ABAP](#).
4. If the application is not shown as active, choose [Create](#) and wait for the subscription to be provisioned.

### 3.3.1.2.8 Creation of Additional Administrator Users (Optional)

After you have set up the ABAP environment for administration, add more administrator users for the ABAP environment.

After you have completed the administrative setup, only your user is available as administrator in the ABAP environment. You will probably want to create more administration users for your ABAP environment. Creating more administration users also helps you verify that your administrative setup is correct and complete.

## Related Information

[Adding a User as Org Manager for the Cloud Foundry Organization \[page 187\]](#)

[Adding a User as Space Manager for the Cloud Foundry Space \[page 188\]](#)

[Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)

[Creating an Employee Record for a New Administrator \[page 189\]](#)

[Assigning the ABAP Environment Administrator Role to the New Administrator User \[page 190\]](#)

### 3.3.1.2.8.1 Adding a User as Org Manager for the Cloud Foundry Organization

After you have created a Cloud Foundry organization for the ABAP environment, you can add more users as organization managers.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#) and entered additional administrators, you can skip these steps.

#### Procedure

1. Log on to the SAP BTP cockpit as administration user for Cloud Foundry.
2. Go to your subaccount for the ABAP environment.
3. From the navigation area, choose [Org Members](#).
4. Choose [Add Members](#).
5. Enter the user's e-mail address in the [E-mails](#) field.
6. Select the checkbox for org manager.
7. Choose [OK](#).

### 3.3.1.2.8.2 Adding a User as Space Manager for the Cloud Foundry Space

After you have created a Cloud Foundry space for the ABAP environment, you can add more users as space managers.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#) and entered users as additional administrators, you can skip these steps.

#### Procedure

1. Log on to the SAP BTP cockpit as administration user for Cloud Foundry.
2. Go to your subaccount for the ABAP environment.
3. Choose [Spaces](#).
4. Choose the Cloud Foundry space in the Cloud Foundry org.
5. In the navigation area, choose [Members](#).
6. Choose [Add Members](#).
7. Enter the user's e-mail address in the [E-mails](#) field.
8. Select the checkbox for space manager.
9. Choose [OK](#).

### 3.3.1.2.8.3 Logging on to the Administration Launchpad of the ABAP Environment

To add another user as administrator for the ABAP instance, log on to the administration launchpad of the ABAP environment.

#### Prerequisites

You are the administrator who has created the ABAP system (see [Creating an ABAP System \[page 180\]](#)).

## Procedure

1. In the SAP BTP cockpit, go to the Cloud Foundry subaccount for the ABAP environment as administrator.
2. Choose the space in the Cloud Foundry subaccount.
3. In the navigation area, choose  Services  Service Instance .
4. Choose the service instance for the ABAP environment.
5. Choose [View Dashboard](#).

The logon window for the administration launchpad of the ABAP environment opens.

6. If you are using the custom Identity Authentication service for SAP BTP, you don't need to enter your user and password, but choose the available link of the custom Identity Authentication service and log on.
7. If you need to enter your user and password, remember to log on with the right user that was defined before (see [Creating an ABAP System \[page 180\]](#)).

## Results

The administration launchpad opens. You can now proceed with creating an employee record for the administrator.

### 3.3.1.2.8.4 Creating an Employee Record for a New Administrator

To be able to create a new administration user, you first need to create an employee record for the new administrator.

## Prerequisites

You have logged on to the administration launchpad of the ABAP environment as administrator user (see [Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)).

## Procedure

1. From the SAP Fiori launchpad, open the [Maintain Employees](#) app.
2. Choose [Create](#).
3. Enter the data for the new administrator, including employee ID, name, and e-mail address.

The entered e-mail is important because it is the default approach to authorization and represents the mapping between the user in the identity provider and in the ABAP environment.

4. Choose *Create*.

## Results

You can now create a business user and assign administration roles to the business user.

### 3.3.1.2.8.5 Assigning the ABAP Environment Administrator Role to the New Administrator User

Add the new administration user as administrator to the ABAP instance.

#### Prerequisites

You have logged on to the administration launchpad of the ABAP environment as administrator (see [Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)). You have created an employee record for another, new administrator.

#### Procedure

1. On the SAP Fiori launchpad, choose the *Maintain Business Users* tile.
2. Choose *New*.
3. In the following dialog, choose the employee record of the new administrator from the list.
4. Confirm the system message that you want to create a new user.
5. Add the time zone to the business user data.
6. On the *Assigned Business Roles* tab, choose *Add*.
7. Select the checkbox for the predefined administrator business role `SAP_BR_ADMINISTRATOR`.
8. Select the checkbox for other administrator roles that are available for your organization and choose *OK*.
9. Choose *OK*.
10. Choose *Save*.

### 3.3.1.2.9 Creation of Developer Users

After you have set up the ABAP environment for development, create developer users for the ABAP environment.

Creating developer users also helps you verify that your development setup is correct and complete.

### 3.3.1.2.9.1 Creating New Space Members and Assigning Space Developer Roles to Them

For your developers, create new members in the space for the ABAP environment and assign developer roles to them.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

#### Procedure

1. Log on to the SAP BTP cockpit as Cloud Foundry administrator.
2. Choose the tile for the Cloud Foundry subaccount.
3. In the navigation area, choose *Spaces*.
4. Choose the space for the ABAP environment.
5. In the navigation area, choose *Members*.

All members currently assigned to the space are shown in a list.

6. Choose [Add Members](#).
7. Enter e-mail addresses of your developer users.
8. You can use commas, spaces, semicolons, or line breaks to separate members.
9. Select the space developer role for the users and save your changes.

### 3.3.1.2.9.2 Creating an Employee Record for a New Developer

Create an employee record for a new developer user for the ABAP environment.

#### Procedure

1. Log on to the administration launchpad of the ABAP environment as administrator (see [Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)).
2. Choose the *Maintain Employees* tile.
3. Choose *Create*.

4. Enter the data for the new developer, including employee ID, name, and e-mail address.

The entered e-mail is important because it represents the mapping between the user in the identity provider and in the ABAP environment.

5. Choose *Save*.

### 3.3.1.2.9.3 Assigning the ABAP Developer User to the ABAP Developer Role

Add the developer user as business user to the ABAP instance.

#### Prerequisites

You have created an employee record for the new developer user (see [Creating an Employee Record for a New Developer \[page 191\]](#)).

You have created an ABAP developer role based on the business role template SAP\_BR\_DEVELOPER, which provides developers with unrestricted read and write access in the ABAP environment. Alternatively, you have created a business role based on a development-related business catalog (see [Business Catalogs for Development Tasks \[page 2548\]](#)).

#### Procedure

1. Log on to the administration launchpad of the ABAP environment as administrator (see [Logging on to the Administration Launchpad of the ABAP Environment \[page 188\]](#)).
2. On the SAP Fiori launchpad, choose the *Maintain Business Users* tile.
3. Choose *New*.
4. In the following dialog, choose the employee record of the new developer user from the list.
5. Confirm the system message that you want to create a new user.
6. On the *Assigned Business Roles* tab, choose *Add*.
7. Select the checkbox for the ABAP developer role that you have created before (see *Prerequisites*).
8. Choose *OK* and save your entries.

### 3.3.1.2.10 Creating a Service Key (Optional)

In the SAP BTP cockpit, you can create a service key for the ABAP system. You can later use this service key, for example, to log on to the ABAP system from ABAP development tools for Eclipse for test purposes.

#### Context

You can use service keys to generate credentials to communicate directly with a service instance. After you have configured the service keys for your service, local clients, apps in other spaces, or entities outside your deployment can access your service with these keys. Using a service key is one option how you can access the service instance of the ABAP environment, for example, as administrator for test purposes. Note, however, that there's also another option for users of ABAP development tools for Eclipse: instead of using an existing service key, developers and other users with a user and password for the ABAP system can also use the project creation wizard in ABAP development tools for Eclipse to log on to the ABAP service instance and use the service key that is automatically provided by the service instance. You can also create service keys for other use cases, such as: [Creating an Inbound Communication Arrangement with Service Key Type Basic](#) and [Configuring the Timeout of the @sap/approuter Component](#).

#### ⓘ Note

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

#### Procedure

1. As Cloud Foundry administration user, log on to the SAP BTP cockpit.
2. Go to your Cloud Foundry subaccount for the ABAP environment.
3. In the navigation area, choose *Spaces*.
4. Choose the tile for the ABAP environment space.
5. From the navigation area, choose *Services* *Instances and Subscriptions*.
6. Choose the service instance of the ABAP system.
7. In the *Service Keys* section of the details area, choose *Create*.
8. Enter a name for the service key, for example, **ABAP Development**.
9. Choose *Save*.
10. Copy the created service key for later reuse.

### 3.3.1.3 Setup of UI Development in SAP Business Application Studio (Optional)

If developers want to use SAP Business Application Studio to create and deploy SAP Fiori application UIs, you need to perform some administrative activities to set it up.

In the Cloud Foundry environment, SAP Business Application Studio is a service that provides a modular development environment, including the development of SAP Fiori applications. As an alternative to SAP Business Application Studio, you can also use SAP Web IDE. We recommend that you use SAP Business Application Studio.

#### ⓘ Note

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

## Prerequisites

For setting up SAP Business Application Studio, you need the following:

- You are a member of the global account.
- You are an organization manager in your Cloud Foundry subaccount (see also [Adding a User as Org Manager for the Cloud Foundry Organization \[page 187\]](#)).
- You have created an ABAP service instance (see also [Creating an ABAP System \[page 180\]](#)).
- You are a security administrator. If you have created a subaccount for the ABAP environment in the Cloud Foundry environment, your user automatically has the security administration role.

## Setting Up SAP Business Application Studio

#### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip the first two steps.

## Do

Set Up SAP Business Application Studio

Subscribe to SAP Business Application Studio

Assign Role Collection to Developers

Create Service Key

Create Service Destination to ABAP System

- [Subscribing to SAP Business Application Studio \[page 196\]](#)
  - [Assigning Permissions for SAP Business Application Studio \[page 196\]](#)
  - [Creating a Service Key for the ABAP System \[page 197\]](#)
  - [Creating a Destination to the ABAP System for SAP Business Application Studio \[page 198\]](#)
1. To enable the creation and deployment of SAP Fiori application UIs, subscribe to SAP Business Application Studio in the same subaccount in which you've created your ABAP service instance (see [Subscribing to SAP Business Application Studio \[page 196\]](#)).
  2. To allow developers to develop applications using SAP Business Application Studio, assign developers to the `Business_Application_Studio_Developer` role collection (see [Assigning Permissions for SAP Business Application Studio \[page 196\]](#)).
  3. Create a service key for your ABAP system, which you will need to set up a service destination (see [Creating a Service Key for the ABAP System \[page 197\]](#)).
  4. To connect your ABAP system with SAP Business Application Studio, set up a destination in the same subaccount in which you have subscribed to SAP Business Application Studio (see [Creating a Destination to the ABAP System for SAP Business Application Studio \[page 198\]](#)).

### 3.3.1.3.1 Subscribing to SAP Business Application Studio

If developers want to use SAP Business Application Studio as development environment for UIs, you need to subscribe to SAP Business Application Studio in your subaccount.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

For more information about SAP Business Application Studio and how to subscribe to it, see also [Subscribe to SAP Business Application Studio](#).

#### Procedure

1. Log on to the SAP BTP cockpit as Cloud Foundry administrator.
2. Choose the tile of the global account of the Cloud Foundry environment.
3. From the global account, navigate to the same subaccount where the ABAP instance is present.
4. From the navigation area, choose [Services](#) [Service Marketplace](#).
5. Search for **Studio**.  
The [SAP Business Application Studio](#) tile is displayed.
6. Choose the application name.
7. Choose [Create](#).

### 3.3.1.3.2 Assigning Permissions for SAP Business Application Studio

Use the available role collection `Business_Application_Studio_Developer` to assign the relevant permissions to your developers to work in SAP Business Application Studio.

#### Context

Developers using SAP Business Application Studio must be assigned to developer roles based on the `Business_Application_Studio_Developer` role template. When you subscribe to SAP Business Application Studio, a role collection containing the relevant developer permissions is automatically created. You must assign the role collection to your developers.

### Note

In the following steps, you can assign a role collection to an individual user. If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

As an alternative, if you are using a custom identity service and identity provider groups, you can also assign the role collection to the identity provider group for your developers. For more information, see [Setup of a Custom Identity Service \[page 202\]](#) and [Mapping a Role Collection to the Identity Provider Group for Developers \[page 207\]](#).

## Procedure

1. In the SAP BTP cockpit, navigate to your subaccount for the ABAP environment.
2. From the navigation area, choose  [Security](#)  [Role Collections](#) .
3. Choose the role collection `Business_Application_Studio_Developer`.
4. Go to the [Users](#) section and choose [Edit](#).
5. Enter the user ID of the user that you want to assign to the role collection.
6. Save your changes.

### 3.3.1.3.3 Creating a Service Key for the ABAP System

Create a service key for the ABAP system, which will later be needed to configure a destination from SAP Business Application Studio to the ABAP system.

### Note

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

You can use service keys to generate credentials to communicate directly with a service instance. After you have configured the service keys for your service, local clients, apps in other spaces, or entities outside your deployment can access your service with these keys. For more information about service keys, see [Creating Service Keys](#).

1. As Cloud Foundry administration user, log on to the SAP BTP cockpit.
2. Go to your Cloud Foundry subaccount for the ABAP environment.
3. In the navigation area, choose [Spaces](#).
4. Choose the tile for the ABAP environment space.
5. From the navigation area, choose  [Services](#)  [Instances](#) .
6. Choose the service instance for the ABAP system.
7. In the [Service Keys](#) section of the details area, choose [Create](#).
8. Enter a service key name, for example, `ABAP Destination`, and save.

9. From the service key, copy the following for later reuse:
  - The URL of the ABAP system from the `url` element
  - The client ID from the `clientid` element in the `uaa` section
  - The content of the `clientsecret` element in the `uaa` section
  - The URL from the `url` element in the `uaa` section

## Related Information

[SAP BTP Cockpit](#)

### 3.3.1.3.4 Creating a Destination to the ABAP System for SAP Business Application Studio

Learn how to set up a destination in the same Cloud Foundry subaccount in which you have subscribed to SAP Business Application Studio to establish communication between the ABAP environment and SAP Business Application Studio.

## Context

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

## Procedure

1. Log on to the SAP BTP cockpit as administrator.
2. From your global account, navigate to your Cloud Foundry subaccount.
3. In the navigation area, choose *Destinations*.
4. Choose *New Destination*.
5. Enter the following data:

Field	User Input
Name	Enter a name for the destination, for example, <code>SAP_Business_Application_Studio</code> .
Type	<code>HTTP</code>
Description	

Field	User Input
<i>URL</i>	Enter the URL of the ABAP system that you copied from the <url> element in the service key (see <a href="#">Creating a Service Key for the ABAP System [page 197]</a> ).
<i>Proxy Type</i>	<i>Internet</i>
<i>Authentication</i>	<i>OAuth2UserTokenExchange</i>
<i>Client ID</i>	Enter the content of the <clientID> element that you copied from the uaa section of the service key (see <a href="#">Creating a Service Key for the ABAP System [page 197]</a> ).
<i>Client Secret</i>	Enter the content of the <clientsecret> element that you copied from the uaa section of the service key (see <a href="#">Creating a Service Key for the ABAP System [page 197]</a> ).
<i>Token Service URL Type</i>	Choose <i>Dedicated</i> .
<i>Token Service URL</i>	Enter <uaa-url>/oauth/token, where <uaa-url> is the content of the <url> element that you copied from the uaa section of the service key (see <a href="#">Creating a Service Key for the ABAP System [page 197]</a> ).

6. Choose *New Property* and add the following properties:

Property	Value
<i>HTML5.DynamicDestination</i>	<b>true</b>
<i>HTML5.Timeout</i>	<b>60000</b>
<i>WebIDEEnabled</i>	<b>true</b>
<i>WebIDEUsage</i>	<b>odata_abap,dev_abap,abap_cloud</b>

7. Make sure that the *Use default JDK truststore* checkbox is checked.
8. Choose *Save*.

## Related Information

[Creating a Destination for Cross-Subaccount Communication \[page 200\]](#)

### 3.3.1.3.5 Creating a Destination for Cross-Subaccount Communication

Learn how to create a destination with SAML assertion authentication to establish communication between an ABAP environment system and SAP Business Application Studio when they are set up in different subaccounts.

#### Procedure

1. Log on to the SAP BTP cockpit as an administrator.
2. From your global account, navigate to your Cloud Foundry subaccount.
3. In the navigation area, choose **Connectivity** **Destinations**.
4. Choose *New Destination*.
5. Enter the following data:

Field	User Input
Name	Enter a name for the destination, for example <b>SAP_Business_Application_Studio</b> .
Type	<b>HTTP</b>
Description	Optionally, enter a description.
URL	Enter the URL of your system by copying the <i>Host Name</i> from the <i>Communications Systems</i> app, for example <b>1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com</b>
Proxy Type	<b>Internet</b>
Authentication	<b>SAMLAssertion</b>
Audience	Enter the URL of your system and add <b>-web</b> as follows <b>1a354373-d200-46f6-9d5c-daab9a65d9b6.abap-web.eu10.hana.ondemand.com</b>
AuthnContextClassRef	<b>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</b>

6. Choose *New Property* and add the following properties:

Property	Value
<b>HTML5.DynamicDestination</b>	<b>true</b>
<b>HTML5.Timeout</b>	<b>60000</b>
<b>WebIDEEnabled</b>	<b>true</b>
<b>WebIDEUsage</b>	<b>odata_abap,dev_abap</b>
<b>nameidFormat</b>	<b>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</b> or <b>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</b>

Property	Value
	depending on the login_attribute configuration chosen during ABAP system provisioning (see <a href="#">Creating ABAP System</a> )

7. Make sure that the *Use default JDK truststore* checkbox is ticked.
8. Choose [Save](#).

## Related Information

[Creating a Destination to the ABAP System for SAP Business Application Studio \[page 198\]](#)  
[Creating a Communication System for SAP Business Application Studio](#)

### 3.3.1.3.5.1 Creating a Communication System for SAP Business Application Studio

Create a communication system to set up communication between your ABAP environment system and SAP Business Application Studio.

## Procedure

1. In the SAP BTP cockpit, download your subaccount-specific trust certificate by navigating to [Connectivity](#) [Destinations](#) [Download Trust](#) .
2. In the SAP Fiori launchpad of your ABAP environment system, create a communication system by using the [Communication Systems](#) app. See [How to Create Communication Systems](#).
3. Navigate to section [General](#) [SAML Bearer Assertion Provider](#) and switch the toggle button to [ON](#).
4. Upload the [Signing Certificate](#) first and then enter the following data:

#### Note

The signing certificate is the trust certificate that you have downloaded from your destination.

Field	User Input
<b>User ID Mapping Mode</b>	User Name
<b>SAML Bearer Issuer</b>	From the <a href="#">Signing Certificate Subject</a> , copy the common name (the string after <b>CN=</b> ) and paste it in the <a href="#">SAML Bearer Issuer</a> field.

### **3.3.1.4 Setup of a Custom Identity Service**

For your business users in the ABAP environment, you set up a custom identity service.

#### **Custom Identity Service for the ABAP Environment**

For platform users such as the administrator, the default identity provider is SAP ID Service, which is set up by default. For the business users in the ABAP environment, you must set up a custom identity service. For example, the users of your deployed application or users of subscribed apps or services, such as SAP Business Application Studio, are business users. For more information about platform and business users in the ABAP environment, see [User Types](#).

In this documentation, a custom trust configuration is described using SAP Cloud Identity Services - Identity Authentication. The Identity Authentication service is a cloud solution for identity lifecycle management for SAP BTP applications. You can set up SAP BTP as a service provider, and Identity Authentication service as an identity provider.

#### **Identity Authentication Service as Proxy**

If you have your own custom identity providers at your company, you can use SAP Cloud Identity Services - Identity Authentication as a proxy. For more information about setting up the Identity Authentication service as a proxy, see [Establish Trust and Federation Between UAA and Identity Authentication](#) and [Configure Trust with Corporate Identity Provider](#).

#### **More Information**

This documentation describes the basic trust concepts and helps you to get started with a simple landscape. To learn more about trust setup with custom identity providers, check the following, more detailed documentation:

- For more information about trust in SAP BTP, see [Trust and Federation with Identity Providers](#).
- For more information about the Identity Authentication service, see [Overview](#) and [Initial Setup](#).

### 3.3.1.4.1 Establishing Trust of Type OpenID Connect

If you want to use a custom identity provider, you must set up trust between the subaccount and the SAP Cloud Identity Services - Identity Authentication service.

#### Context

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

If you have custom identity provider, you can use a function in SAP BTP cockpit to set up trust between your subaccount and the Identity Authentication service for SAP BTP automatically. The trust configuration is of type *OpenID Connect*.

For more information, see [Establish Trust and Federation Between SAP Authorization and Trust Management Service and Identity Authentication](#).

#### Procedure

1. In the SAP BTP cockpit, go to the subaccount for your ABAP system.
2. From the navigation area, choose [Security](#) [Trust Configuration](#).
3. Choose [Establish Trust](#).
4. In the following popup, select a identity provider from the dropdown list.

Only identity providers that are associated with your customer ID are shown.

5. Choose [Establish Trust](#).  
Trust of type *OpenID Connect* between your subaccount and the identity provider is generated.
6. Log on to the Identity Authentication service.
7. From the navigation area, choose [Applications & Resources](#) [Applications](#).
8. Search for the application that has been created as part of the trust setup.

The name of the application has the format *SAP BTP subaccount <Subaccount Name>*, but you can change it if needed.

##### ⓘ Note

Older application names have the format *XSUAA\_<Subaccount Name>*.

9. Verify that the subject name identifier matches the login\_attribute chosen during ABAP system provisioning.

ABAP login_attribute	SAP Cloud Identity Services Subject Name Identifier
email	email
user_name	login name

### 3.3.1.4.2 Creating Identity Provider Users and Groups

Learn how you create identity provider users and groups in the Identity Authentication service.

#### Context and Use Cases

This configuration is needed for easier developer authentication using Identity Authentication service. With the creation of identity provider groups, you can also more easily map role collections to entire user groups instead of single users, or map identity provider groups to groups in the Neo environment, for example.

This documentation illustrates the basic concepts and helps you to get started with users and groups in a simple landscape for the ABAP environment.

#### Prerequisites

You have already created developer users in the ABAP environment (see [Creation of Developer Users \[page 190\]](#)).

### 3.3.1.4.2.1 Creating an Identity Provider Group for Developers

In the administration console of the Identity Authentication service, you can create an identity provider group for the developers.

#### Context

An identity provider group for developers is useful, for example, if you want to assign a role collection to all developers in one step, such as the *Application\_Studio* role collection for SAP Business Application Studio.

You also use identity provider groups for developers if you want to assign all developers to a corresponding group in the Neo environment, which is necessary if your developers want to use SAP Web IDE as UI development tool.

For more information about creating user groups in the Identity Authentication service, see [Create a New User Group](#).

## Procedure

1. Log on to the tenant's administration console for the Identity Authentication service at <https://<tenant ID>.accounts.ondemand.com/admin> as administration user.
2. In the navigation area, choose [Users & Authorizations](#) [User Groups](#).
3. If no user group for developers exists, choose [+ Add](#).
4. Enter a name, for example, **Developers**.
5. Enter a display name and a description.
6. Choose [Save](#).
7. Note down the group name for later reference.

### 3.3.1.4.2.2 Creating Identity Provider Users for Developers

In the administration console of the Identity Authentication service, you can create identity provider users for the developers working in the ABAP environment.

## Context

You create identity provider users if you work with your own custom identity provider, such as the Identity Authentication service for SAP BTP. You can use the identity provider users, for example, for developers who develop UIs and services using SAP Web IDE or SAP Business Application Studio.

For more information about creating users in the Identity Authentication service, see [Create a New User](#).

## Procedure

1. Log on to the tenant's administration console for the Identity Authentication service at <https://<tenant ID>.accounts.ondemand.com/admin> as administration user.
2. In the navigation area, choose [Users & Authorizations](#) [User Management](#).
3. Choose [+ Add User](#).
4. Enter the name, e-mail address and login name of the developer.
5. Leave the account type set to **Employee**.
6. Make sure that the account activation is set to **Send activation e-mail**.
7. Choose [Save](#).

### 3.3.1.4.2.3 Assigning the Identity Provider User Group to the Identity Provider Users for Developers

Assign the identity provider user group for developers to the identity provider users that you have created for the developers.

#### Prerequisites

You have already created identity provider users. You have also already created identity provider user groups, for example, for SAP Web IDE, for SAP Business Application Studio, or for both.

#### Context

You now assign these identity provider user groups to the identity provider users for the developers.

For more information about assigning user groups, see [Assign Groups to a User](#).

#### Procedure

1. Log on to the tenant's administration console for the Identity Authentication service at <https://<tenant ID>.accounts.ondemand.com/admin> as administration user.
2. In the navigation area, choose [Users & Authorizations](#) [User Management](#).
3. Choose the identity provider user that you have created for the developer.
4. Choose the [User Group](#) tab.
5. Choose [Assign Groups](#).
6. Choose the identity provider group for developers that you have created (for example, [Developers](#) for SAP Web IDE, or [BAS\\_DEVELOPERS](#) for SAP Business Application Studio).
7. Choose [Save](#).

### 3.3.1.4.2.4 Mapping a Role Collection to the Identity Provider Group for Developers

For developers to be able to work with SAP Business Application Studio, you can use a predefined role collection and assign it to the identity provider group that you created for the developers.

#### Prerequisites

##### ⓘ Note

If you have run the booster [Prepare an Account for ABAP Development](#), you can skip this step.

You have subscribed to SAP Business Application Studio (see [Subscribing to SAP Business Application Studio \[page 196\]](#)).

#### Context

##### ⓘ Note

These steps are only relevant if developers want to use SAP Business Application Studio as development environment for UIs.

Developers who want to load, develop, and deploy applications in SAP Application Studio must be assigned to developer roles based on the [Business\\_Application\\_Studio\\_Developer](#) role template. There's no need to create these roles for SAP Business Application Studio. When you subscribe to SAP Business Application Studio, role collections together with these business role templates are created automatically.

This procedure briefly describes how to assign the role collections to the developer user group in your custom identity provider. For more information about managing authorizations, see [Manage Authorizations](#).

#### Procedure

1. Log on to the SAP BTP cockpit as Cloud Foundry administrator.
2. Choose the tile of the global account of the Cloud Foundry environment.
3. In the SAP BTP cockpit, navigate to your subaccount for the ABAP environment.
4. From the navigation area, choose [Security](#) [Role Collections](#) .
5. Choose the [Business\\_Application\\_Studio\\_Developer](#) role collection.
6. Go to the [User Groups](#) section and choose [Edit](#).
7. Select the identity provider where the user group is stored.
8. Enter the name of the user group that you created before, for example, [Developers](#).

9. Save your entries.

### 3.3.1.5 Setup of UI Development in SAP Web IDE (Optional)

If your developers want to use SAP Web IDE as a development tool for creating UIs, you as an administrator need to perform some additional settings.

If your developers want to use SAP Web IDE, which is part of the Neo environment, they need to work in the Cloud Foundry and the Neo environments. To enable an easier authentication of the developers in the Cloud Foundry and Neo environments, set up SAP BTP as a service provider, and Identity Authentication service as an identity provider. For the integration, you must set the trust on both sides and map the identity provider group for developers to its corresponding Neo group.

#### → Recommendation

We recommend that you use SAP Business Application Studio as a UI development environment instead of SAP Web IDE.

### 3.3.1.5.1 Neo Environment: Trust Setup

To enable an easier authentication of the developers in the Cloud Foundry and Neo environments, set up SAP BTP as a service provider, and Identity Authentication service as an identity provider.

Before you follow the steps outlined here for setting up trust between the Neo application and the Identity Authentication service, make sure that you have already set up the Identity Authentication service as custom identity provider for the Cloud Foundry account. For more information, see [Setup of a Custom Identity Service \[page 202\]](#).

#### 3.3.1.5.1.1 Exporting SAML Identity Provider Metadata

Export the SAML metadata of the identity provider for the SAML trust setup on the service provider side.

#### Context

You need this step only if the SAML metadata file for the Identity Authentication service tenant has not been downloaded already. For more information about tenant SAML 2.0 configuration, see [Tenant SAML 2.0 Configuration](#).

## Procedure

1. Log on to the tenant's administration console for the Identity Authentication service at `https://<Your custom Identity Authentication service tenant>.accounts.ondemand.com/admin` as administration user.
2. In the navigation area, choose  *Applications & Resources*  *Tenant Settings*.
3. Choose *SAML 2.0 Configuration*.
4. Choose *Download Metadata File* (or download it directly from `https://<Your custom Identity Authentication service tenant>.accounts.ondemand.com/saml2/metadata`).
5. Specify a file name and save it locally to your hard drive.

 Note

Specify a file name that you can easily recognize later, for example, `metadataIDP<tenant-id>.xml`, because you need to download and upload multiple SAML metadata files later.

### 3.3.1.5.1.2 Configuring the Neo Account as SAML Service Provider and Export SAML Metadata

Configure the Neo account as local SAML service provider.

## Context

 Note

The following steps are only relevant if your developers will use SAP Web IDE in the Neo environment for UI development.

This configuration is needed for developer authentication in the Neo environment (Web IDE runs in the Neo environment).

In the SAML 2.0 communication, each account acts as a service provider. In this documentation, we use the term "local service provider" to describe the account in SAP BTP as a service provider in the SAML 2.0 communication. You need to configure how the local service provider communicates with the identity provider. This includes, for example, setting a signing key and certificate to verify the service provider's identity and encrypt data.

For more information, see [Configure SAP BTP as a Local Service Provider](#).

## Procedure

1. Log on to the SAP BTP cockpit as Neo administrator.
2. Go to the subaccount for the Neo environment.
3. In the SAP BTP cockpit, choose **Security** **Trust**.
4. On the *Local Service Provider* tab, choose *Edit*.
5. Change the configuration type from *Default* to *Custom*.
6. If the fields for the signing key and the signing certificate are already filled, leave them as they are and don't generate a new key pair.
7. If the fields for the signing key and the signing certificate are empty, choose *Generate Key Pair*.
8. Set principal propagation to *Enabled*.
9. Leave the settings for force authentication as they are.
10. Choose *Save* and confirm the alert message.
11. If you haven't downloaded the SAML metadata file yet: Choose *Get Metadata*.

The metadata file is downloaded automatically.

12. Rename the downloaded file so that you can easily recognize it later on, for example, to `metadataNeo<tenant-id>.xml`,

You must download and upload multiple SAML metadata files. Renaming the file helps you distinguishing the metadata files from each other.

## Results

You can now switch to the *Application Identity Provider* tab to import the SAML identity provider metadata.

### 3.3.1.5.1.3 Establishing Trust to the SAML Identity Provider for the Neo Account

Import the SAML identity provider metadata to the Neo account to configure SAML trust to the identity provider (that is, the Identity Authentication service tenant).

## Prerequisites

You have downloaded the SAML identity provider metadata file from the Identity Authentication service tenant (see [Exporting SAML Identity Provider Metadata \[page 208\]](#)).

## Context

### ⓘ Note

The following steps are only relevant if your developers will use SAP Web IDE in the Neo environment for UI development.

You want to use Identity Authentication service as an SAML 2.0 identity provider. For more information, see [Configure Trust to the SAML Identity Provider](#).

## Procedure

1. Log on to the SAP BTP cockpit as Neo administrator.
2. In the SAP BTP cockpit, navigate to the subaccount for the Neo environment.
3. In the navigation area, choose .
4. Choose the *Application Identity Provider* tab.
5. Choose *Add Trusted Identity Provider*.
6. Choose *Browse*.
7. Upload the metadata file of the custom Identity Authentication service tenant.

### ⓘ Note

During the setup of the ABAP environment, you need to upload multiple SAML metadata files. Make sure that you upload the correct file. If you have followed the naming conventions suggested in this documentation, the file needed here is `metadataIDP<tenant-id>.xml`.

8. Enter a description.
9. Choose *Save*.

## 3.3.1.5.1.4 Creating a Neo Application as Trusted Service Provider

Create and configure an application to represent the Neo account in the Identity Authentication Service tenant as SAML 2.0 service provider.

### ⓘ Note

The following steps are only relevant if your developers will use SAP Web IDE in the Neo environment for UI development.

## Creating an SAML Application for the Neo Account

As a first step, you need to create the application for the Neo account.

1. Log on to the tenant's administration console for the Identity Authentication service at `https://<tenant ID>.accounts.ondemand.com/admin` as administration user.
2. In the navigation area, choose  *Application & Resources*  *Applications*.
3. Choose *+ Add*.
4. Enter an application name for the Neo account (the service provider) in the Identity Authentication service.
5. Choose *Save*.

## Configuring the Neo Application as Trusted Service Provider

Import the SAML service provider metadata from the Neo environment to the Identity Authentication service tenant (that is, the SAML identity provider). This configures SAML trust to the Neo account as SAML service provider.

1. Under *Applications*, choose the entry for the Neo application that you have just created.
2. Choose *SAML 2.0 Configuration*.
3. Choose *Browse*.
4. Upload the metadata file of the SAML service provider from the Neo account that you have downloaded before (see also [Configuring the Neo Account as SAML Service Provider and Export SAML Metadata \[page 209\]](#)).

### Note

During the setup of the ABAP environment, you need to upload multiple SAML metadata files. Make sure that you upload the correct file. If you have followed the naming conventions suggested in this documentation, the file needed here is `metadataNeo<tenant-id>.xml`.

5. Choose *Save*.

## Configuring the Subject Name Identifier Sent to Cloud Foundry

Configure the subject name identifier that the Identity Authentication service (as identity provider) sends to the Neo application. Use the email address as user identifier for the Neo account.

1. Under *Applications*, choose the entry for the Neo application.
2. Choose *Subject Name Identifier*.
3. Choose *Basic Configuration* and select *E-Mail* from the dropdown list.
4. Choose *Save*.

## Configuring the Default Name ID Format for the Neo Application (Optional)

Optionally, you can configure the default name ID attribute of the identity provider for Neo. Use the email address as user identifier for the Neo account.

1. Under [Applications](#), choose the entry for the Neo application in the Identity Authentication service.
2. Choose [Default Name ID Format](#).
3. Choose the [E-Mail](#) radio button.
4. Choose [Save](#).

## Configuring the SAML Assertion Attributes for Cloud Foundry

Include the attributes `first_name`, `last_name`, `mail`, and `Groups` for the SAML assertion.

1. Under [Applications](#), choose the entry for the Neo application.
2. Choose [Attributes](#).
3. Choose [Add](#) to enter the following attributes (if they don't already exist):

Name	Value
<code>first_name</code>	<i>First Name</i>
<code>last_name</code>	<i>Last Name</i>
<code>mail</code>	<i>E-Mail</i>
<code>Groups</code>	<i>Groups</i>

### ⓘ Note

Make sure that you enter **Groups** (with upper case **G**), not **groups**, in the **Name** field, so that it matches the assertion attribute used for the Neo environment.

4. Choose [Save](#).

## More Information

[Configuring Applications](#)

[Configure Trust](#)

[Configure the Subject Name Identifier Sent to the Application](#)

### 3.3.1.5.2 Enabling SAP Web IDE Full-Stack in the Neo Account

Enable the SAP Web IDE Full-Stack service in the Neo account if it is not yet enabled.

#### Context

##### ⓘ Note

You only need this step if your developers want to use SAP Web IDE as environment for UI development. As an alternative, you can also use SAP Business Application Studio.

SAP Web IDE Full-Stack is a development environment to create new or extend existing SAP Fiori, SAPUI5, or hybrid applications. For more information about the SAP Web IDE Full-Stack, see [SAP Web IDE Full-Stack](#).

#### Procedure

1. Log on to the SAP BTP cockpit for the Neo environment as Neo administration user.
2. In the navigation area, choose *Services*.
3. Scroll down to the *Web IDE Full-Stack* tile.
4. If the service is not yet enabled, choose the *Web IDE Full-Stack* tile.
5. On the *Service: Web IDE Full-Stack: Overview* screen, choose *Enable*.
6. Wait for the status to change to the green *Enabled* status.

This may take a few minutes.

### 3.3.1.5.3 Creation of Developer Users for SAP Web IDE

If your developers want to use SAP Web IDE, use identity provider users for them to ease log on to the Neo environment.

If you have followed the steps for setting up a custom identity provider, you have already created a developer group in the tenant for the Identity Authentication service (see also [Setup of a Custom Identity Service \[page 2021\]](#)). Now you need to create a developer group in the Neo environment and map this group to the developer group in the tenant for the Identity Authentication service.

### 3.3.1.5.3.1 Creating a Neo Group for Developers

Define a Neo group to map the developer group of the identity provider to the `DiDeveloper` role of SAP Web IDE.

#### Context

##### ⓘ Note

The following steps are only relevant if your developers will use SAP Web IDE in the Neo environment for UI development.

For more information about roles, see [Managing Roles](#).

#### Procedure

1. Log on to the SAP BTP cockpit (Neo account) as Neo administration user.
2. In the navigation area, choose [Security](#) [Authorizations](#).
3. Choose the [Groups](#) tab.
4. Choose [New Group](#).
5. Enter a group name, for example, **Developers**, and choose [Save](#).
6. In the [Roles](#) screen area, choose [Assign](#).
7. As subaccount, select the service provider account of the Web IDE, for example, **sapwebide EU1**.
8. As application, select the Web IDE application (**di**).
9. As role, select the predefined Web IDE developer role **DiDeveloper**.
10. Choose [Save](#).

### 3.3.1.5.3.2 Mapping the Identity Provider Group Developers to the Neo Group

Map the developer group of the identity provider to the Neo developer group with the Web IDE `Developer` role.

#### Context

##### ⓘ Note

The following steps are only relevant if your developers will use SAP Web IDE in the Neo environment for UI development.

The mapping is part of the trust configuration to the SAML identity provider. For more information, see [Configure Trust to the SAML Identity Provider](#).

#### Procedure

1. Log on to the SAP BTP cockpit (Neo account) as Neo administration user.
2. In the navigation area, choose [Security](#) [Trust](#).
3. Choose the [Application Identity Provider](#) tab.
4. Choose the link for the custom identity authentication service tenant.
5. Choose the [Groups](#) tab.
6. If no group for Neo developers exists, choose [Add Assertion-Based Group](#).
7. Enter the following data:

Option	Description
<i>Group</i>	Select <a href="#">Developers</a> .
<i>Mapping Rules (part 1)</i>	Enter <a href="#">Groups</a> .
<i>Mapping Rules (part 2)</i>	Select <a href="#">equals</a> .
<i>Mapping Rules (part 3)</i>	Enter the developer group name from the Identity Authentication service that was created by the administrator for the Identity Authentication service, for example, <a href="#">Developers</a> .  If you are not the responsible administrator for the Identity Authentication service, make sure that you are informed about the group name (see also <a href="#">Creating an Identity Provider Group for Developers [page 204]</a> ).

8. Choose [Save](#).

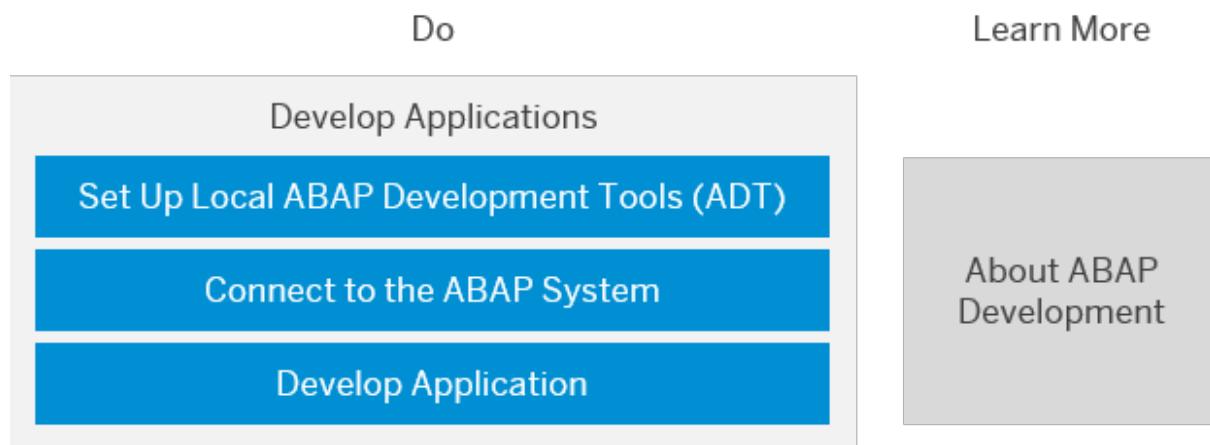
### 3.3.1.6 Getting Started as a Developer in the ABAP Environment

Learn about your first steps to get started as a developer in the ABAP environment.

#### ⓘ Note

This documentation informs you about the first steps as a developer and assumes that the necessary steps for setting up the ABAP environment are done. For more information about how to set up the ABAP environment and getting started as an administrator, see [Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#).

## Developing Applications: Overview



- <https://tools.hana.ondemand.com/#abap> [https://tools.hana.ondemand.com/#abap]
- [Connect to the ABAP System \[page 708\]](#)
- <https://developers.sap.com/tutorials/abap-environment-gcts.html#b1afe95c-9acb-4f75-8f80-1abe93cb524f> [https://developers.sap.com/tutorials/abap-environment-gcts.html#b1afe95c-9acb-4f75-8f80-1abe93cb524f]
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html]
- <https://help.sap.com/viewer/DRAFT/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/59656c2f858748fe976456248d390c5c.html> [https://help.sap.com/viewer/DRAFT/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/59656c2f858748fe976456248d390c5c.html]
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html]

1. Download and install ABAP Development Tools (ADT) from <https://tools.hana.ondemand.com/#abap>. See [Video Tutorial: Configure ABAP Development Tools](#).

 **Note**

SAP GUI is not supported in the ABAP environment. You can only use ADT as your development environment.

2. Create an ABAP cloud project with ADT to connect to the ABAP system in the ABAP environment (see [Connect to the ABAP System \[page 708\]](#)).
3. Develop your application (see [Development in the ABAP Environment](#)). To learn more about how to develop applications, see [Tutorial: Create Your First ABAP Console Application](#) and [Video Tutorial: Create Application](#).

## More Information About Development in the ABAP Environment

For more information about ABAP development, see [Development in the ABAP Environment](#).

### 3.3.1.7 Tutorial Overview

## Getting Started

### Mission

Build an SAP Fiori app Using the ABAP RESTful Application Programming Model

- [Create Database Table and Generate UI Service](#)
- [Enhance the Business Data Model and Enable OData Streams](#)
- [Enhance the Business Object Behavior With Unmanaged Internal Numbering](#)
- [Enhance the Business Object Behavior With Determinations](#)
- [Enhance the Business Object Behavior With Validations](#)
- [Enhance the Business Object Behavior With Instance Action](#)
- [Enhance the Business Object Behavior With Factory Action](#)
- [Enhance the Business Object Behavior With Dynamic Feature Control](#)
- [Write an ABAP Unit Test for the RAP Business Object](#)

*Group: Create and Expose CDS Views (Unmanaged Scenarios)*

- [Create a Simple Database Table for ABAP Environment](#)
- [Create and Expose Core Data Services Based on a Database Table](#)
- [Add Transactional Behavior to Your Core Data Services](#)

*Group: Create an ABAP Core Data Services (CDS) View in SAP BTP, ABAP Environment*

- Enhance an ABAP Core Data Services (CDS) View in SAP Business Technology Platform (BTP) ABAP Environment

## ABAP Dictionary and ABAP Language

Group: Start Developing ABAP Tables

- Create an ABAP Database Table and Relevant ABAP Dictionary Objects
- Create ABAPDoc Comments in Your Class in ABAP Environment
- Create an ABAP Managed Database Procedure (AMDP) and Analyze Its Performance

## Analytics

- Develop and Consume Queries on SAP Analytics Cloud
- Develop Queries Based on Booking Supplement and Consume on SAP Analytics Cloud

## Authorization Model

Group: Create Authorization Model with SAP BTP, ABAP Environment

- Create Data Model and Lock Object in SAP BTP, ABAP Environment
- Create Behavior and Service in SAP BTP, ABAP Environment
- Create Authorization Model and App in SAP BTP, ABAP Environment
- Maintain Business Roles and Access Restrictions in SAP BTP, ABAP Environment

## Business Configuration

Group: Create a SAP Fiori based Table Maintenance App

- Create Business Configuration Maintenance Object
- Providing Authorization Control for a Business Configuration Maintenance Object
- Use Custom Business Configurations app
- Explore additional features of the Business Configurations app

## Lifecycle Management and CI/CD

- Transfer Your ABAP Source Code With SAP Cloud Platform ABAP Environment via abapGit
- Create Branches with SAP BTP, ABAP Environment
- Use Checkout Functionality of Branches to Switch Between Different Versions of Code
- Transport a Software Component Between two Systems
- Use abapGit to Transfer ABAP Source Code to the Cloud
- Connect ABAP On-Premise to a Central Check System for Custom Code Migration Using ABAP Test Cockpit (ATC)

## Cloud Operations

- Monitor an SAP BTP, ABAP Environment Service Using SAP Focused Run (FRUN)
- Monitor an SAP BTP, ABAP Environment Service Using SAP Cloud ALM (CALM)

## Eventing

- How to Create RAP Business Events in SAP BTP, ABAP Environment
- Event Consumption in ABAP Development Tools for Eclipse
- How to Create RAP Business Events in an On-Premise System (In Progress)

## Integration

*Group:* Expose an ABAP CDS View Externally Using a Communication Scenario

- Expose a Standard Core Data Service for ABAP Environment
- Maintain a Communication Arrangement for Inbound Communication

*Mission:* Get Data From an On-Premise System Using a Remote Function Call (RFC) 

- Get Data from an On-Premise System Using a Remote Function Call (RFC) Scenario
- Remote Function Call (RFC) - Connect to Your On-Premise System from SAP BTP, ABAP Environment
- Test the Business API (BAPI) with an ABAP Console App
- Get Data from an On-Premise System Using a Custom Entity
- RFC: Inspect Your Class in the ABAP Debugger
- RFC: Enhance the Handler Class With Filtering

*Mission:* Get Data from a Remote System Using an OData Service

- Enable Connection From a Remote Instance of SAP BTP, ABAP Environment
- Prepare Consuming System and Create Service Consumption Model

- Create Remote Client Proxy and ABAP Console App 
- Implement Custom Entity and Query Implementation Class 

*Mission:* Connect to SAP S/4HANA Cloud with SAP BTP, ABAP Environment 

- Create Communication System to Connect to SAP BTP, ABAP Environment 
- Create Service Consumption Model for Business Partner and Sales Order Item Cube 
- Create Service Binding for Bonus Plan Scenario with SAP BTP, ABAP Environment 

without group/mission:

- Call an External API and Parse the Response in SAP BTP, ABAP Environment 
- Create RFC Proxy Class 
- Connect ABAP On-Premise to a Central Check System for Custom Code Migration Using ABAP Test Cockpit (ATC) 
- Using ODBC driver for SQL Service 
- Consume SOAP Based Web Services with SAP BTP, ABAP Environment 

## Tools, User, Identity Management

- Create an SAP Cloud Platform ABAP Environment Trial User 

without group/mission:

- Install ABAP Development Tools (ADT) and abapGit Plugin 
- Create a Developer User in SAP Cloud ABAP Environment 
- Provision Users into your SAP BTP, ABAP Environment 

## UI Development

- Develop and Run SAP Fiori Application With SAP Business Application Studio 
- Add the Shopping Cart Fiori Application to FLP 
- Add SAP BTP ABAP environment App to Portal Site 
- Developing and extending SAP Fiori Elements Apps 
- RAP Feature Showcase App 
- Create a Travel App with SAP Fiori Elements Based on OData V4 RAP Service 

without group/mission:

- Develop and Run SAP Fiori Application With Visual Studio Code 
- Develop SAP Fiori App & Deploy to Cloud Foundry Using SAP-managed App Router 
- Add Cloud Foundry to Portal Side 
- Set SAP BTP ABAP Environment as a ContentProvider for SAP Build Workzone, standard edition 
- Use Custom Themes in the SAP BTP, ABAP Environment 

## SAP S/4HANA Cloud, ABAP Environment

Group: Develop an SAP Fiori App to Trigger Purchase Requisitions API 

- Create a Shopping Cart Business Object 
- Create Value Help, Enhance the Behaviour Definition and Behaviour Implementation of the Shopping Cart Business Object 
- Integrate released Purchase Requisition API into Online Shop Business Object 
- Develop a Custom UI with SAP S/HANA Cloud ABAP Environment 
- Integrate List Report into Fiori Launchpad 

### Without group/mission

- Implement a Business Add-In (BAdI) to Check a Purchase Requisition 
- Implement a Field Control Using a Business Add-In (BAdI) 
- Create Custom Logic Using Key User Extensibility and Perform Trace 
- Create and Debug a BAdI Implementation in ABAP Development Tools (ADT) 

## Badges

Taster: [Get Started with SAP BTP, ABAP Environment](#) 

Level 1: [Get to know the ABAP RESTful Application Programming Model](#) 

Level 2: [Level up with SAP BTP, ABAP Environment](#) 

Level 3: [SAP BTP, ABAP Environment: Intermediate Topics](#) 

### 3.3.2 Getting Started with Custom Code Analysis in the ABAP Environment

Learn how to get started with Custom Code Analysis in the SAP BTP, ABAP environment.

You want to use the SAP BTP, ABAP environment to analyze your custom code remotely in your SAP ECC or SAP S/4HANA systems? This documentation will help you gain insight into all the options to work with the ABAP Test Cockpit (ATC) in the SAP BTP, ABAP environment.

The following list covers all possible use cases and will guide you to the respective documentation chapters providing more details on each use case.

1. **ABAP Test Cockpit Configurator:** Use this app to maintain ATC configurations using the SAP Fiori launchpad.  
For more information on this app, see [ABAP Test Cockpit Configurator](#).  
To learn how to enable the app, see [Enable Usage of the ABAP Test Cockpit Configurator App](#).
2. **Custom Code Analysis:** If you want to check your custom code in your on-premise system remotely using the SAP BTP, ABAP environment as the central check system, you need to set up the Custom Code Migration app on the SAP Fiori launchpad.

To enable the Custom Code Migration app, see [Enable Usage of the Custom Code Migration App](#).  
To learn all about custom code migration, see the [Custom Code Migration Guide for SAP S/4HANA 2023](#).

3. **ATC Developer Scenario:** You can use a system in the SAP BTP, ABAP environment as a central check system to run ATC checks from an on-premise system against this system. To do this, a number of configurations will be necessary.  
To learn how to use the ATC as a central check system, see [Configuring Remote ATC Using a Central Check System](#).  
To learn how to set up the ATC developer scenario, see [Configuring the ATC Developer Scenario](#).
4. **Working with Exemptions:** Learn about your options to exempt your ATC findings.  
To learn how to enable the Approve ATC Exemptions app to be able to exempt the findings from your custom code analysis, see [Enable Usage of the Approve ATC Exemptions App](#).  
For more information on the Approve ATC Exemptions app, see [Approve ATC Exemptions](#).
5. **Schedule Custom Code Analysis:** The Schedule Custom Code Analysis app enables you to schedule projects created in the Custom Code Migration app as application jobs. With this functionality, the analyses can be performed once or periodically.  
To learn more about the app and how to enable it, see [Schedule Custom Code Analysis](#).
6. **Maintain the Baseline:** In the Custom Code Migration app, it is now possible to maintain the ATC baseline. This means that you can add/remove custom code project check results to/from the baseline and choose whether you would like to exempt your findings, suppress your findings or reduce their priority. After adding findings to the baseline, they won't appear in the subsequent check results of the custom code project.  
To learn more about the baseline functionality in the Custom Code Migration app, see [Custom Code Migration](#).

## Related Information

[SAP BTP, ABAP environment](#)  
[Custom Code Analysis \[page 2614\]](#)

## 3.4 Getting Started in the Kyma Environment

As an administrator, you must perform several steps to set up a fully operational Kyma environment to which you can connect the chosen SAP solutions.

### Prerequisites

- To perform administrative and development tasks, you need a global account and one or several subaccounts for which you can provision the Kyma environment. For details, see [Create a Kyma Instance \[page 2992\]](#).

- The subaccount must have the following entitlements assigned:
  - Kyma environment (added by the subaccount manager)
  - Service Manager (added by default). However, if you removed the entitlement, you must add it again in your subaccount

For details, see [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#).

## Procedure

1. Set up a Kubernetes cluster with the project "Kyma" to connect and extend SAP systems: [Create a Kyma Instance \[page 2992\]](#)
2. Add the modules you'd like to use: [Kyma Modules \[page 74\]](#).
3. Assign the roles to users to allow the administrators to manage Kyma and the developers to create Functions: [Assign Roles in the Kyma Environment \[page 3137\]](#)

### ⚠ Caution

Assign the roles before the users start using the Kyma dashboard. Not granting the roles results in an error.

4. Use the Kyma environment to integrate external systems: [Register an SAP Customer Experience System](#)

You have the following options:

- Integrate a CX system and the Kyma environment, so that the Functions you develop can use the system's API and receive business events.
- Integrate an SAP S/4 HANA Cloud system to use the services it provides to extend your applications.

After you've integrated externals systems, the developers can start working on their Functions.

5. Extend SAP S/4HANA Cloud by developing event-driven extensions and applications: [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment](#)
6. Group several solutions into one formation to meet the requirements of your business scenario: [Including SAP Systems in a Formation](#)

## Results

You have set up the Kyma environment and connected it as required.

## Next Steps

After the administrator has set up the Kyma environment, the developers can access it through the Kyma dashboard. After logging on, developers can start creating extensions for the SAP systems either from the Kyma dashboard or from the terminal after downloading the kubeconfig file with the cluster configuration. For details, see [Development in the Kyma Environment \[page 1864\]](#).

## 3.4.1 Getting Started with an Enterprise Account in the Kyma Environment

Quickly get started with an enterprise account in the Kyma environment using the SAP BTP cockpit.

### Procedure

1. Get a global account.

After you've signed up for your enterprise account, you get an e-mail with a link to SAP BTP and the logon data as administrator for the global account.

Log on to SAP BTP and navigate to your global account.

#### ⓘ Note

For more information on available plans, including free tier, see [Available Plans in the Kyma Environment \[page 2998\]](#).

2. Set up your subaccount. For more information, see [Create a Subaccount \[page 2173\]](#).

You can create more subaccounts to break down your account model and structure it according to your development scenario. But first it's important you understand how to navigate to your accounts using the SAP BTP cockpit.

You can also download and use the SAP BTP command line interface (btp CLI) to create new subaccounts. See [Download and Start Using the btp CLI Client \[page 2323\]](#) and [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#).

3. Create your Kyma environment. For more information, see [Create a Kyma Instance \[page 2992\]](#).
4. Configure your Kyma environment.

Now that you've set up your account model, it's time to think about member management. You can add members at different levels:

- For Kyma-specific role assignment, see [Assign Roles in the Kyma Environment \[page 3137\]](#).
- To learn more about how user roles are assigned in SAP BTP, have a look at [User and Member Management \[page 104\]](#).

5. Develop your applications and extend SAP solutions.

Check out [Development in the Kyma Environment \[page 1864\]](#) to learn more.

### Related Information

[Available Plans in the Kyma Environment \[page 2998\]](#)

## 3.4.2 Getting Started with a Trial Account in the Kyma Environment

Quickly get started with a trial account in the Kyma environment.

### Procedure

1. Get a global account.

Sign up for a free trial account. See [Get a Free Trial Account \[page 144\]](#). For more information about the scope of our trial offering, see [Trial Accounts and Free Tier \[page 82\]](#).

For more information on available plans, including free tier, see [Available Plans in the Kyma Environment \[page 2998\]](#).

2. Set up your subaccount.

When you register for a trial account, a subaccount is created for you.

If you want to create more subaccounts to break down your account model and structure it according to your development scenario, see [Create a Subaccount \[page 2173\]](#). But first it's important you understand how to navigate to your accounts using the SAP BTP cockpit.

You can also download and use the SAP BTP command line interface (btp CLI) to create new subaccounts. See [Download and Start Using the btp CLI Client \[page 2323\]](#) and [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#).

3. Create your Kyma environment. For details, see [Create a Kyma Instance \[page 2992\]](#) and [Enable Trial Kyma Environment \[page 229\]](#).

4. Configure your Kyma environment.

Now that you've set up your account model, it's time to think about member management. You can add members at different levels:

- For Kyma-specific role assignment, see [Assign Roles in the Kyma Environment \[page 3137\]](#).
- To learn more about how user roles are assigned in SAP BTP, have a look at [User and Member Management \[page 104\]](#).

5. Develop your applications and extend SAP solutions.

Check out [Development in the Kyma Environment \[page 1864\]](#) to learn more.

You can also provide extensions for the SAP Commerce Cloud, SAP Cloud for Customer, and SAP Field Service Management systems. For more information, read [Extending SAP Solutions \[page 2022\]](#).

### 3.4.2.1 About the Trial Account

A trial account lets you try out the Kyma environment for free with a restricted use of the platform resources and services. Access is open to everyone. Trial accounts are intended for personal exploration, and not for production use or team development.

#### Trial Period

The trial account is valid for 90 days. During this time, you can explore and use the basic functionality of SAP BTP, Kyma runtime for 14 days. After 90 days, your account is automatically deleted, and you'll no longer be able to access your data. See [Trial Accounts and Free Tier](#) to learn more about the trial periods.

Note that while your trial account is valid for 90 days, the Kyma environment that you create with it expires **14 days** after its creation and is then deleted permanently. To create a new Kyma environment, you must first disable the expired Kyma on your subaccount. To learn how to do that, see [Disable Expired Trial Kyma Environment \[page 230\]](#).

##### ⚠ Caution

Once your account has expired, your cluster is deleted and all the cluster resources are removed. We recommend that you back up your cluster configuration to restore it quickly if you choose to create a new Kyma environment.

#### Scope and Limitations

- It's possible to provision only one Kyma environment per global account.
- You can manage members in your trial account.
- Kyma trial accounts are available in all SAP BTP regions.
- A trial account provides you with a 1-node cluster with 4 vCPU and 16 GB of memory. An idle cluster without any customer workload uses around 33% of vCPU and 33% of memory.
- A trial account comes with a cluster with built-in eventing.
- Events are processed in-memory and are not persisted to disk. Not delivered events will be lost on restart of the eventing infrastructure.
- There's no service level agreement with regards to the availability of the platform.

#### Related Information

[Trial Accounts and Free Tier \[page 82\]](#)

[Assign Roles in the Kyma Environment \[page 3137\]](#)

## 3.4.2.2 Setting Up Your Trial Account

Your global trial account is set up automatically, so that you can start using it right away. However, if one or more of the automatic steps fail, you can also finalize the setup manually by following the steps below.

### 3.4.2.2.1 Create Your Trial Subaccount

The first thing that is needed in the setup of your trial account is the creation of a subaccount. If this step was successful in the SAP BTP cockpit, you can directly skip to the next section.

#### Procedure

1. Navigate into your global account by choosing [Enter Your Trial Account](#).
2. Choose [New Subaccount](#).
3. Configure your trial account:

Field	Input
Display Name	<code>trial</code>
① Note	<p>The value for this parameter is a sample one, you can provide a name of your choice.</p>
Description	Optional
Provider	Desired infrastructure provider
Region	Desired region
Subdomain	<code>&lt;your_id&gt;trial</code> Example: <code>P0123456789trial</code>
Enable beta features	Optional Enables the use of beta services and applications.
Custom Properties	Optional You can use custom properties to identify and organize the subaccounts in your global account. For example, you can filter subaccounts by custom property in some cockpit pages.

## Results

You have successfully set up your trial subaccount.

### 3.4.2.2.2 Add Entitlements

Once you have a subaccount (whether it was created automatically or you followed the steps described above), you need entitlements to get your trial up and running.

#### Procedure

1. Navigate to your subaccount by choosing its tile.
2. Choose *Entitlements* from the left-hand side navigation.
3. Choose *Configure Entitlements* to enter edit mode.
4. Choose *Add Service Plans* and select all the service plans available for your subaccount.

#### ⓘ Note

To select a service plan, choose a service from the left and tick all the service plans that appear on the right. Do that for all services.

5. Once you've added all the service plans, you see them all in a table. Before you choose Save, for all the plans with numerical quota, choose **+** to increase the amount to the maximum (until the icon becomes disabled).
6. Finally, choose *Save* to save all your changes and exit edit mode.

### 3.4.2.2.3 Enable Trial Kyma Environment

After you've successfully added the entitlements, you can enable your trial Kyma environment and start developing.

#### Procedure

1. Navigate to your subaccount.
2. Create your Kyma environment. For details, see [Create a Kyma Instance \[page 2992\]](#).
3. Provide the name of your cluster and, optionally, a description.

Wait until the cluster is provisioned and you see the link to the Kyma dashboard.

4. Access the Kyma dashboard. For further information on the development options, see [Development in the Kyma Environment](#) [page 1864].

## Results

Your trial Kyma environment is ready to use.

### 3.4.2.2.4 Disable Expired Trial Kyma Environment

After your trial Kyma environment has expired, you must disable it to remove all the data connected to the cluster from SAP systems.

#### Procedure

1. Navigate to your subaccount.
2. In the [Kyma Environment](#) section of your subaccount overview, click [Disable Kyma](#).

#### ⓘ Note

On deletion of the expired cluster, we attempt to delete the Service Instances that you created with the cluster. If we cannot do that, you have to [remove the Service Instances yourself](#) before you disable such a [Kyma Environment](#).

### 3.4.2.2.5 Extend Trial Kyma Environment

You cannot extend your trial Kyma environment. After your trial Kyma environment has expired, you must disable it on your subaccount and enable it again to create a new cluster.

#### Procedure

1. Navigate to your subaccount.
2. [Disable your expired Kyma environment](#). [page 230]
3. After your Kyma environment has been successfully disabled, [create a new Kyma environment](#). [page 229]

# 4 Development

Develop and run business applications on SAP Business Technology Platform (SAP BTP) using our cloud application programming model, APIs, services, tools, and capabilities.

SAP BTP supports multiple environments. It features a number of tools and programming languages that increase your flexibility and freedom when developing applications. You can choose the environment that best suits your use cases and skill set and get to work. You also have the possibility to integrate applications you've built with other solutions by SAP.

Environment options

Cloud Foundry	Kyma	ABAP
<ul style="list-style-type: none"><li>• Simplified developer experience for business application development</li><li>• Large choice of programming languages</li><li>• Intuitive “code-to-container” packaging and deployment, managed by the platform</li><li>• Platform-managed application security patching and updates</li><li>• Automatic application routing, load balancing, health checks, and multilevel self-healing</li><li>• Support for CAP – an opinionated business app development framework</li></ul>	<ul style="list-style-type: none"><li>• Take full advantage of the advanced features and rich ecosystem of Kubernetes</li><li>• Free choice of programming languages and models (containerized deployments)</li><li>• Combines microservices and serverless functions</li><li>• Built-in, managed service mesh based on Istio, and other cloud-native open-source modules to reduce the development effort</li><li>• Built-in, managed event mesh</li><li>• Managed infrastructure: day-2 operations, security patches, and updates</li><li>• Full administrator access</li><li>• Refined horizontal and vertical automatic scalability</li><li>• Dedicated application runtime</li><li>• Zero downtime infrastructure setup by default</li><li>• Support for CAP – an opinionated business app development framework</li><li>• Support for on-premise connectivity</li></ul>	<ul style="list-style-type: none"><li>• ABAP programming language</li><li>• Fast prototyping with ABAP RESTful Programming Model (RAP)</li><li>• Integrated development lifecycle</li><li>• Reuse existing on-prem ABAP assets</li></ul>

Cloud Foundry	Kyma	ABAP
<a href="#">Comparison: SAP BTP, Kyma Runtime and SAP BTP, Cloud Foundry Runtime</a>	<a href="#">Comparison: SAP BTP, Kyma Runtime and SAP BTP, Cloud Foundry Runtime</a>	<a href="#">Development in the ABAP Environment</a>
<b>Benefits</b>	<ul style="list-style-type: none"> <li>• No infrastructure vendor lock-in</li> <li>• Build scalable multitenancy business applications (SaaS)</li> <li>• Out-of-the-box consumption of SAP and hyperscaler services</li> <li>• Built on industry standards and open technology</li> </ul>	
<b>Goals</b>	<ul style="list-style-type: none"> <li>• Managed build-on approach</li> <li>• Enterprise-grade business applications and services</li> <li>• Cloud-native web applications and services</li> <li>• Scalable, microservice-based applications</li> <li>• Small to medium extensions built with CAP/low-code tooling</li> </ul>	<ul style="list-style-type: none"> <li>• Open build-on approach</li> <li>• Enterprise-grade applications</li> <li>• Cloud-native development of apps and services</li> <li>• Low latency infra-services communication</li> <li>• Reduced infrastructure management effort</li> <li>• Highly scalable, microservice-based applications</li> <li>• Applications built with CAP</li> </ul>
<b>Approaches</b>		<ul style="list-style-type: none"> <li>• User-centric process extensions</li> <li>• Robust, transactional cloud applications</li> <li>• Migrating and adapting add-ons to the cloud</li> <li>• Reusing existing on-premise ABAP code</li> <li>• Enabling ABAP developers to go to the cloud</li> </ul>

Cloud Foundry	Kyma	ABAP
<ul style="list-style-type: none"> <li>S Any major programming languages</li> <li>k SAP Fiori/UI5 and SAP HANA</li> <li>i</li> <li>I</li> <li>I</li> <li>s</li> </ul>	<ul style="list-style-type: none"> <li>• Kubernetes knowledge</li> <li>• Docker</li> <li>• NodeJS or Python for serverless functions</li> <li>• Any major programming language</li> <li>• SAP Fiori/UI5 and SAP HANA</li> </ul>	<ul style="list-style-type: none"> <li>• Ability to write modern ABAP code</li> <li>• Core data services</li> <li>• SAP Fiori/UI5 and SAP HANA</li> </ul>

We provide detailed information about developing, configuring, and deploying your applications depending on your preferred environment and development approach.

#### [Development in the Cloud Foundry Environment \[page 233\]](#)

Learn more about developing applications on the SAP BTP, Cloud Foundry environment.

#### [Development in the ABAP Environment \[page 705\]](#)

Learn more about developing applications in the ABAP environment.

#### [Development in the Kyma Environment \[page 1864\]](#)

Learn more about developing applications in SAP BTP, Kyma runtime.

## Related Information

[SAP BTP Administrator's Guide](#)

[Developing with the SAP Cloud Application Programming Model \[page 243\]](#)

[Consuming Services in SAP BTP](#)

## 4.1 Development in the Cloud Foundry Environment

Learn more about developing applications on the SAP BTP, Cloud Foundry environment.

### Overview

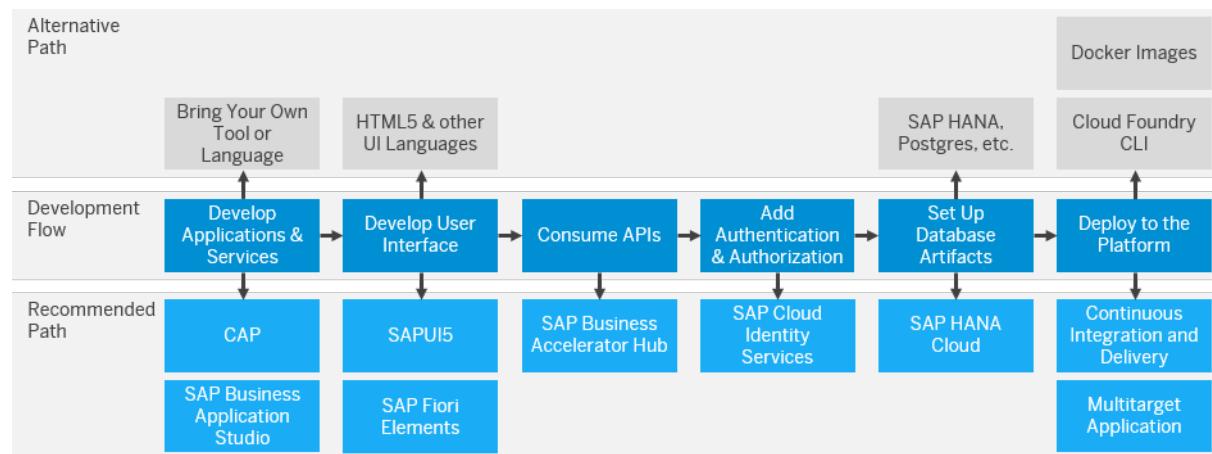
SAP BTP, Cloud Foundry environment is an open Platform-as-a-Service (PaaS) targeted at microservice development and orchestration.

<b>Develop polyglot applications</b>	Build on open standards with SAP Java, Node.js, and Python buildpacks or bring your own language with community buildpacks for PHP, Ruby, Go.
<b>Manage the lifecycle of applications</b>	Start, stop, scale, and configure distributed cloud applications using standard Cloud Foundry tools, our web-based administration user interface for SAP BTP, and dev-ops capabilities.

<b>Optimize development and operations</b>	Use the rich set of SAP BTP services including messaging, persistence, and many other capabilities, such as built-in security, compliance, elastic scale, and high-availability setup.
<b>Use the application programming model</b>	Use programming languages, libraries, and APIs tailored for full-stack application development.
<b>Manage Cloud Foundry orgs and spaces</b>	Create and delete Cloud Foundry orgs and spaces and add members. Create, assign, and change space quota plans.
<b>Use built-in cloud-native capabilities</b>	Utilize cloud-native principles built into the platform, such as containerization and multitenancy, to create resilient, portable, and efficient applications.

The following graphic is designed to help you find the information you need for your programming purposes. The bottom row represents the tools, frameworks, services, and deployment options recommended by SAP. If you want full flexibility you can also bring your own development tools and languages, as shown in the top row.

## Development Options Overview



- [Development Languages \[page 260\]](#)
- [Developing HTML5 Applications in the Cloud Foundry Environment \[page 391\]](#)
- [Developing SAP HANA in the Cloud Foundry Environment \[page 567\]](#)
- [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#)
- [Deploy Docker Images in the Cloud Foundry Environment \[page 703\]](#)
- [Developing Applications and Services \[page 238\]](#)
- [Developing User Interface \[page 387\]](#)
- [Consuming APIs \[page 498\]](#)
- [Adding Authentication and Authorization \[page 499\]](#)
- [Setting Up Database Artifacts \[page 567\]](#)

- [Deploying to the Cloud Foundry Environment \[page 569\]](#)
- [Developing with the SAP Cloud Application Programming Model \[page 243\]](#)
- [SAP Business Application Studio \[page 239\]](#)
- [Developing SAPUI5 \[page 388\]](#)
- <https://api.sap.com/> [https://api.sap.com/]
- [Protecting Your Application \[page 501\]](#)
- [SAP HANA Cloud \[page 568\]](#)
- [Continuous Integration and Delivery \[page 698\]](#)
- [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#)

## The Recommended Path

This development approach offers guidance for important development decisions and features proven best practices recommended by SAP. You can follow a model path for application and service development that is based on the Cloud Application Programming Model (CAP). When working with CAP, we recommend using Java and Node.js because they receive the highest level of tool support and are well suited for most use cases. This path provides you with a list of key aspects to consider, but the order shown in these steps isn't mandatory. You can adapt the steps as you wish to better fit your use case.

## Choose Your Own Path

Because of the polyglot nature of the Cloud Foundry environment, you're also free to choose your own approach. You're not forced to use one language exclusively, but can choose between Java, Node.js, and Python. Concerning tools you can either use the Cloud Foundry Command Line Interface (CF CLI) or other tools to develop and deploy your application. You're also free to decide if you want to develop and deploy your applications in the multitarget format (MTA).

For more information on the supported programming languages, see:

- [Developing Java in the Cloud Foundry Environment \[page 261\]](#)
- [Developing Node.js in the Cloud Foundry Environment \[page 343\]](#)
- [Developing Python in the Cloud Foundry Environment \[page 354\]](#)

If you already have monolithic applications running on SAP BTP and are looking for a way to run them in the Cloud Foundry environment, read our migration best practice guide. See [Migrating from the Neo Environment to the Multi-Cloud Foundation for SAP BTP \(Cloud Foundry and Kyma\)](#).

## Related Information

[Cloud Foundry Environment \[page 49\]](#)

[Troubleshooting App Deployment and Health](#) ↗

[SAP BTP, Cloud Foundry Runtime](#)

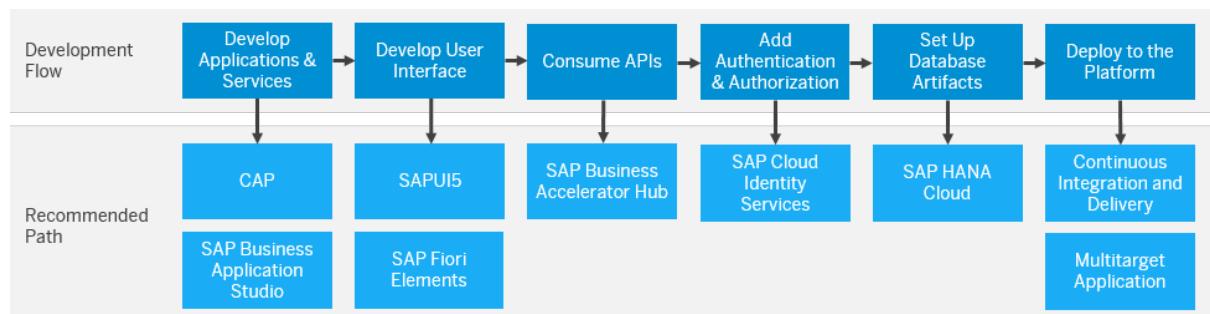
## 4.1.1 Best Practices

Choose the development environment, tools, APIs, and programming model that best suit your needs with recommendations from SAP.

Navigating the large portfolio of solutions, services, tools, and frameworks can be difficult. That's why we want to provide you with recommendations based on our years of experience in helping customers achieve their goals. The best practices listed here are targeted at developers. If you're looking for recommendations on general development planning and setting up SAP BTP, see [SAP BTP Administrator's Guide](#). If you're interested in extending existing applications, see [Extensions \[page 1989\]](#).

The following graphic offers an overview of the most important considerations to take when starting development on SAP BTP for the Cloud Foundry environment. The steps are recommendations, you're free to complete them in a different order if you feel it suits your needs better.

## Development Options



- [Developing Applications and Services \[page 238\]](#)
- [Developing User Interface \[page 387\]](#)
- [Consuming APIs \[page 498\]](#)
- [Adding Authentication and Authorization \[page 499\]](#)
- [Setting Up Database Artifacts \[page 567\]](#)
- [Deploying to the Cloud Foundry Environment \[page 569\]](#)
- [Developing with the SAP Cloud Application Programming Model \[page 243\]](#)
- [Developing SAPUI5 \[page 388\]](#)
- <https://api.sap.com/> [https://api.sap.com/]
- [Protecting Your Application \[page 501\]](#)
- [SAP HANA Cloud \[page 568\]](#)
- [SAP Business Application Studio \[page 239\]](#)
- [Continuous Integration and Delivery \[page 698\]](#)
- [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#)

## The Recommended Path: The SAP Cloud Application Programming Model

We recommend using the SAP Cloud Application Programming Model (CAP) for full-stack development. CAP is a framework of languages, libraries, APIs, and tools that guide developers along a proven path of best practices. It was designed with a business domain focus in mind, relying on common patterns and reuse models for programming. With CAP you can develop multitarget applications or automate tasks such as authorization, integration, or localization to make applications and services easier to fix and maintain. CAP is compatible with any development environment, but we recommend SAP Business Application Studio.

For more information, see [Developing with the SAP Cloud Application Programming Model \[page 243\]](#) and [SAP Business Application Studio](#).

## Multitarget Applications

One of the challenges of programming in a cloud environment is the deployment and management of applications consisting of multiple, interdependent components. The agility, flexibility, and resilience of cloud applications brings with it an increased complexity. For example, your application could be targeted at multiple runtimes or consist of interconnected modules created in different tools and programming languages.

To reduce this complexity, we recommend programming multitarget applications (MTA). That means packaging all the components of your application into a single archive file. Doing so makes managing the application's lifecycle easier and enables you to automate processes, for example through the Continuous Integration and Delivery service.

For more information, see [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#).

## SAP Alert Notification Service

Applications running in the Cloud Foundry environment are constantly monitored through health checks. If an application doesn't respond, for example because it crashed, it fails the health check and is automatically restarted. Because the system restarts applications quickly to avoid major disruptions, it can be difficult to identify underlying problems. For more information about configuring Health Checks, see <https://docs.cloudfoundry.org/devguide/deploy-apps/healthchecks.html>

To support the smooth operation of your applications and services, we recommend using the SAP Alert Notification Service. Using a standardized environment-agnostic model, the service collects any update from the application checks as well as crucial technical information from other services on SAP BTP. It also handles custom scenarios that only occur in your specific application and environment. Each piece of information is translated into a common event model.

You can subscribe to events that are of interest to you and use a delivery channel of your choice, for example email or a custom webhook allowing you to send events to any REST API endpoint in the public internet. SAP Alert Notification Service also natively supports integration with external systems, such as Slack, Microsoft Teams, VictorOps, ServiceNow. For more information on events, see [SAP Alert Notification Service Events](#).

## Related Information

[What Is SAP Alert Notification Service for SAP BTP](#)

### 4.1.2 Developing Your First Application

Follow any of these tutorials to develop your first application on SAP BTP. Choose your preferred programming model and technology.

Name	Technology
<a href="#">Build a Business Application Using CAP for Node.js</a>	SAP Cloud Application Programming Model (CDS and Node.js)
<a href="#">Develop a Simple Hello World Application Using SAPUI5</a>	SAPUI5
<a href="#">Create an Application with SAP Java Buildpack</a>	Java
<a href="#">Create an Application with Cloud Foundry Python Buildpack</a>	Python
<a href="#">Create an Application with Cloud Foundry Node.js Buildpack</a>	Node.js

### 4.1.3 Developing Applications and Services

Get started with developing applications and services in the Cloud Foundry environment.

## Basic Considerations

At the outset of every development project stand questions about the goal you want to achieve, which skills, tools, and programming languages are required, and how the finished application or service is deployed and maintained. This topic provides an overview of the programming models, development languages, tools, and services you can leverage in the SAP BTP, Cloud Foundry environment to help you address those questions.

## SAP Business Application Studio

Learn about the SAP Business Application Studio, the environment tailored for efficient development of business applications. For more information, see [SAP Business Application Studio \[page 239\]](#).

## **Business Application Pattern**

To find information about the recommended development architecture for business applications in the SAP BTP, Cloud Foundry environment, see [Business Application Pattern \[page 240\]](#).

## **SAP Cloud Application Programming Model**

The SAP Cloud Application Programming Model (CAP) is the recommended framework for application and service development in the Cloud Foundry environment. To learn more, see [Developing with the SAP Cloud Application Programming Model \[page 243\]](#).

## **Services in the Cloud Foundry Environment**

In a PaaS environment, external dependencies such as databases, messaging systems, and file systems are services. To use a service, you need to create a service instance and bind it to your application, using either the cockpit or the Cloud Foundry Command Line Interface (CF CLI). To learn more, see [Using Services in the Cloud Foundry Environment \[page 248\]](#).

## **Development Languages**

Find information and tutorials for the main development languages supported in the SAP BTP, Cloud Foundry environment: Java, Node.js, and Python. See [Development Languages \[page 260\]](#).

## **Multitenant Applications**

If you want to make your applications and services available for use to other SAP BTP users, you can design them as multitenant applications. This allows consumers, who are called tenants in this scenario, to access what you've created through dedicated URLs. For more information, see [Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#).

### **4.1.3.1 SAP Business Application Studio**

SAP Business Application Studio is a new cloud service in the SAP BTP, Cloud Foundry environment offering a modern development environment tailored for efficient development of business applications.

SAP Business Application Studio provides a desktop-grade IDE experience in the cloud, including command line functionalities and optimized editors.

It features dev spaces that function like isolated virtual machines. They contain tailored tools and pre-installed runtimes that can save you time and effort when setting up your development environment and that allow you to efficiently develop, test, build, and run your solutions both locally or in the cloud.

SAP Business Application Studio is the recommended environment when working with the Cloud Application Programming Model (CAP).

For more information, see [SAP Business Application Studio](#).

### 4.1.3.2 Business Application Pattern

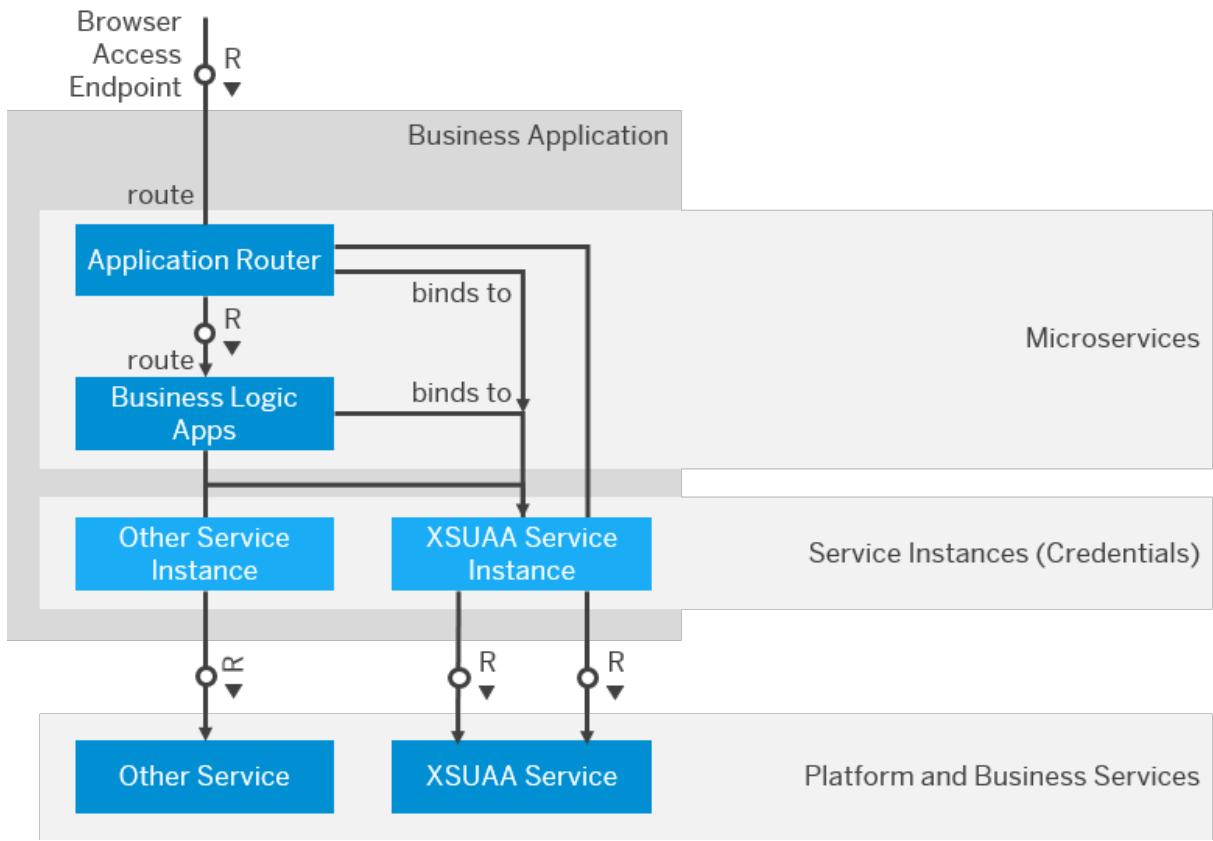
In the Cloud Foundry environment, SAP is promoting a pattern for building applications. We use the term **Business Application** to distinguish from single applications in the context of the Cloud Foundry environment.

This topic covers the following:

- [Application Patterns \[page 240\]](#)
- [Application Router \[page 241\]](#)
- [UAA Service \[page 242\]](#)
- [Platform and Business Services \[page 242\]](#)
- [Application Deployment \[page 242\]](#)

### Application Patterns

The diagram below is a logical view, abstracting from numerous details like the CF router, CF controller, and represents architecture of a business application. In general, a business application consists of multiple microservices which are deployed as separate applications to SAP BTP, Cloud Foundry environment.



Microservices, service instances, bindings, services, and routes are entities known to the platform.

Create Microservices from "pushing" code and binaries to the platform resulting in a number of application instances each running in a separate container.

Services are exposed to apps by injecting access credentials into the environment of the applications via service binding. Applications are bound to service instances where a service instance represents the required configuration and credentials to consume a service. Services instances are managed by a service broker which has to be provided for each service (or for a collection of services).

Routes are mapped to applications and provide the actual application access points / URLs.

A business application is a collection of microservices, service instances, bindings, and routes, which together represent a usable web application from an end user point of view. These microservices, services instances, bindings, and routes are created by communicating with the CF / XSA Controller (for example, using a command line interface).

SAP provides a set of libraries, services, and component communication principles, which are used to implement (multi-tenant) business applications according this pattern.

## Application Router

Business applications embracing a microservice architecture, consist of multiple services that can be managed independently. Still this approach brings some challenges for application developers, like handling security in a consistent way and dealing with same origin policy.

Application router is a separate component that addresses some of these challenges. It provides three major functions:

- Reverse proxy - provides a single entry point to a business application and forwards user requests to respective backend services
- Serves static content from the file system
- Security – provides security-related functionality like logon, logout, user authentication, authorization, and CSRF protection in a central place

The application router exposes the endpoint accessed by a browser to access the application.

## UAA Service

The User Account and Authentication (UAA) service is a multi-tenant identity management service, used in the SAP BTP, Cloud Foundry environment. Its primary role is as an OAuth2 provider, issuing tokens for client applications to use when they act on behalf of the users of the Cloud Foundry environment. It can also authenticate users with their credentials for the Cloud Foundry environment, and can act as an SSO service using those credentials (or others). It has endpoints for managing user accounts and for registering OAuth2 clients, as well as various other management functions.

## Platform and Business Services

The platform provides a number of backing services like SAP HANA, MongoDB, PostgreSQL, Redis, RabbitMQ, Audit Log, Application Log, and so on. Also, the platform provides various business services, like retrieving currency exchange rates. In addition, applications can use user-provided services, which are not managed by the platform.

In all these cases applications can bind and consume required services in a similar way. See [Services Overview documentation](#) for general information about services and their consumption.

## Application Deployment

There are two options for a deployer (human or software agent), how to deploy and update a business application:

- Native deployment: Based on the native controller API of the Cloud Foundry environment, the deployer deploys individual applications and creates service instances, bindings, and routes. The deployer is responsible for performing all these tasks in an orchestrated way to manage the lifecycle of the entire business application.
- [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#) (MTA): Based on a declarative model the deployer creates an MTA archive and hands over the model description together with all application artifacts to the SAP Deploy Service. This service is performing and orchestrating all individual (native) steps to deploy and update the entire business application (including blue-green deployments).

### 4.1.3.3 Developing with the SAP Cloud Application Programming Model

The SAP Cloud Application Programming Model (CAP) is a framework of languages, libraries, and tools for building enterprise-grade services and applications. It guides developers along a ‘golden path’ of proven best practices and a great wealth of out-of-the-box solutions to recurring tasks.

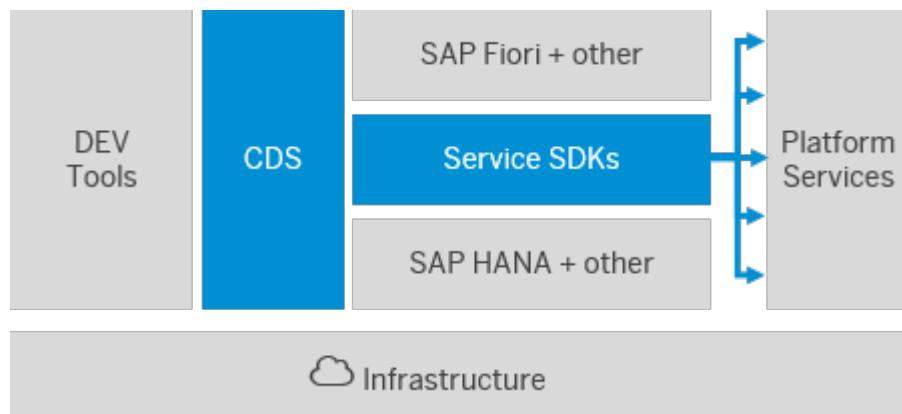
Use Core Data & Services (CDS) to build data models and service definitions on a conceptual level. These CDS models are used as inputs for the data, service, and UI layers. They’re then translated to native artifacts, for example SQL database schemas, and interpreted to automatically serve requests at runtime.

In summary, CDS is used as a business level data definition source, and to generate the artifacts at the persistence layer. It’s used to define visual aspects relating to the data, with those definitions (annotations) defining the UI layer. And it’s used to generate the application service layer.

We provide seamless integration with the Cloud Foundry environment on SAP Business Technology Platform. This makes it easier for you to deploy your application and consume platform services.

The programming model is compatible with any development environment, but we recommend using the SAP Business Application Studio.

The following graphic shows the relationship between CAP, SAP Business Technology Platform, platform services, and development tools:



#### [Best Practices \[page 244\]](#)

To help you create concise and comprehensible models with CDS, we have put together a list of best practices.

#### [Core Data and Services \(CDS\) Language Reference Documentation \[page 244\]](#)

CDS is the backbone of the SAP Cloud Application Programming Model (CAP). It provides the means to declaratively capture service definitions and data models, queries, and expressions in plain (JavaScript) object notations. CDS features to parse from a variety of source languages and to compile them into various target languages.

#### [Developing Business Applications Using Java \[page 245\]](#)

The CAP Java SDK enables developing SAP Cloud Application Programming Model (CAP) applications in Java. While the SAP Business Application Studio provides excellent support to develop CAP Java applications, you can also develop locally with your tool of choice, for example Eclipse.

#### [Developing Business Applications Using Node.js \[page 246\]](#)

Develop a sample business service using Core Data & Services (CDS), Node.js, and SQLite, by using the SAP Cloud Application Programming Model (CAP). Develop on your local environment and deploy to the Cloud.

#### [References \[page 247\]](#)

Find additional references in the following sections.

### 4.1.3.3.1 Best Practices

To help you create concise and comprehensible models with CDS, we have put together a list of best practices.

- [Domain Modeling](#) - Find here an introduction to the basics of domain modeling with CDS, complemented with recommended best practices.
- [Providing Services](#) - Learn in this guide how to define services exposed through REST and OData APIs, including an overview of generic features to serve metadata as well as most CRUD requests out of the box, input validations and more.
- [Consuming Services](#) - CAP provides manifold generic handlers for common, recurring tasks, which serve many requests out-of-the-box, thereby capturing proven best practices collected from a wealth of successful SAP applications.
- [Using Databases](#) - This guide is about defining, providing, implementing, deploying, and publishing services — so about Service Providers in general.
- [Localization](#) - Guides you through the steps to internationalize your application to provide localized versions with respect to both Localized Models as well as Localized Data.
- [Localized Data](#) - Guides you through the steps to internationalize your application to provide localized versions.
- [Temporal Data](#) - Temporal data allow maintaining information relating to past, present, and future application time.
- [Authorization](#) - About restricting access to data by adding respective declarations to CDS models, which are then enforced in service implementations.
- [SaaS Extensibility](#) - How SaaS subscribers can extend SaaS applications on a tenant level.
- [Multitenancy](#) - How to implement multitenant aware applications (SaaS applications) with CAP.
- [Serving OData APIs](#) - All about OData Annotations, OData Vocabularies, Data Aggregation, and OData V2 Support.
- [Serving Fiori UIs](#) - Explains how to add one or more SAP Fiori Elements apps to a CAP project and how to add SAP Fiori Elements annotations to respective service definitions.
- [Deploy to Cloud](#) - How to deploy your services to the cloud.
- [Native SAP HANA](#) - How to use existing database objects with CDS.

### 4.1.3.3.2 Core Data and Services (CDS) Language Reference Documentation

CDS is the backbone of the SAP Cloud Application Programming Model (CAP). It provides the means to declaratively capture service definitions and data models, queries, and expressions in plain (JavaScript) object

notations. CDS features to parse from a variety of source languages and to compile them into various target languages.

CDS models are plain JavaScript objects complying to the Core Schema Notation (CSN), an open specification derived from JSON Schema. You can easily create or interpret these models, which foster extensions by 3rd-party contributions. Models are processed dynamically at runtime and can also be created dynamically.

For further information, please refer to the following sections:

- [Upgrade to Compiler v2](#) - CDS compiler version 2 (cv2) brings numerous improvements, which allow us to significantly streamline model processing going forward.
- [Definition Language \(CDL\)](#) - A reference and overview of all CDS concepts and features in the form of compact examples.
- [Schema Notation \(CSN\)](#) - Specification of CSN, CDS' canonical format for representing CDS models as plain JavaScript objects, similar to JSON Schema.
- [Query Language \(CQL\)](#) - Documents the CDS Query Language (aka CQL) which is an extension of the standard SQL SELECT statement.
- [Query Notation \(CQN\)](#) - Specification of the Core Query Notation (CQN) format that is used to capture queries as plain JavaScript objects.
- [Expression Notation \(CXN\)](#) - Specification of the Core Expression Notation (CXN) used to capture expressions as plain JavaScript objects.
- [Built-in Types](#) - List of provided built-in types.
- [Common Types & Aspects](#) - Introduces `@sap/cds/common` a prebuilt CDS model shipped with `@sap/cds` that provides common types and aspects.
- [Common Annotations](#) - A reference and glossary of common annotations intrinsically supported by the CDS compiler and runtimes.
- [Compiler Messages](#) - This page lists selected error messages and explanations on how to fix them.
- [The Nature of Models](#) - Introduces the fundamental principles of CDS models.
- [Node.js API](#) - The CDS compiler is implemented in Node.js. Find here the reference documentation for the respective APIs which can be used in CLI tools as well at runtime in CDS service implementations.

### 4.1.3.3 Developing Business Applications Using Java

The CAP Java SDK enables developing SAP Cloud Application Programming Model (CAP) applications in Java. While the SAP Business Application Studio provides excellent support to develop CAP Java applications, you can also develop locally with your tool of choice, for example Eclipse.

The CAP Java SDK supports lean application design by its modular architecture, that means you pick the required features and add them to your application dependencies on demand.

It enables local development by supporting in-memory or file-based SQLite databases. At the same time, the CAP Java SDK enables switching to a productive environment, using, for example, SAP HANA as a database, easily by simply switching the application deployment configuration.

The following sections help you get started.

- [Getting Started](#)
- [Using Local Development](#)

- Starting a New Project 
- Stack Architecture 
- Services 
- Event Handlers 
- Working with Data 
- Building CQL Statements 
- Executing CDS QL Statements 
- Introspecting CQL Statements 
- Working with CDS Models 
- Application Services 
- Fiori Drafts 
- Indicating Errors 
- Request Contexts 
- ChangeSet Contexts 
- Security 
- Remote Services 
- Messaging 
- Multitenancy 
- Persistence Services 
- Audit Logging 
- Development 
- Observability 
- Migration 

#### 4.1.3.3.4 Developing Business Applications Using Node.js

Develop a sample business service using Core Data & Services (CDS), Node.js, and SQLite, by using the SAP Cloud Application Programming Model (CAP). Develop on your local environment and deploy to the Cloud.

The following sections describe how to set up a development environment to get you started.

- Getting Started in a Nutshell 
- Jumpstarting Projects 
- Defining Domain Models 
- Defining Services 
- Using Databases 
- Adding/Serving UIs 
- Adding Custom Logic 
- Deploying to Cloud 
- Model Processing in Node.js 
- The `cds` Facade Object 
- Node.js Core Services API 

- [Authentication](#)
- [Application Service Providers](#)
- [Remote Services](#)
- [Transaction Management](#)
- [Databases](#)
- [Messaging](#)
- [Best Practices](#)
- [Using TypeScript](#)
- [CDS Design Time APIs](#)

### 4.1.3.3.5 References

Find additional references in the following sections.

#### 4.1.3.3.5.1 What's New

Find out what is new and what has changed in CAP since the last release.

- [Release Notes](#)
- [Changelog](#)

#### 4.1.3.3.5.2 Learning Content

Learn about the golden path that CAP gives users to create service-based applications. Get to know the principles that enable you to focus on your domain and avoid boilerplate code. With CAP you focus on what you can do best: Solving business problems.

Try out the [Learning Sources](#) we collected on this page.

#### 4.1.3.3.5.3 Sample Projects

CAP enables you to quickly create business applications by allowing you to focus on your domain logic. It offers a consistent end-to-end programming model that includes languages, libraries, and tools tailored for full-stack development on SAP BTP.

Get started with samples and reusable packages created based on CAP for [Node.js](#) and [Java](#) provided can be run in a local setup on, for example, a SQLite database.

#### 4.1.3.3.5.4 Tutorials

- Learn how to create a business service using CAP for Java and deploy it to SAP BTP: [Build a Business Application Using CAP for Java](#).
- Learn how to create a business service with CDS using Node.js and deploy it to SAP BTP: [Create a Business Service with Node.js using Visual Studio Code](#).
- Find more tutorials using the SAP Cloud Application Programming Model (CAP) in the [Tutorial Navigator](#).

#### 4.1.3.3.5.5 Community, Blogs & Q&A

Learn more about CAP by visiting SAP Community.

- Find more samples, openSAP course material, podcasts, and other related resources in the [SAP Cloud Application Programming Model community](#).
- Read the [blogs](#) including the [original introduction](#) by Daniel Hutzel and a great collection of [starting points](#) by DJ Adams.
- Ask [questions](#) related to CAP to get help from developers and other community members.

#### 4.1.3.3.5.6 Troubleshooting

If you run into issues working with CAP samples or tutorials, please refer to the [Troubleshooting](#) guide. There you can find a collection of frequently asked questions and provided solutions.

Furthermore, you can get support as mentioned in section [Resources](#).

### 4.1.3.4 Using Services in the Cloud Foundry Environment

Learn more about using services in the Cloud Foundry environment, how to create (user-provided) service instances and bind them to applications, and how to create service keys.

- [About Services \[page 249\]](#)
- [Binding Service Instances to Applications \[page 253\]](#)
- [Creating Service Instances \[page 250\]](#)
- [Creating Service Keys \[page 255\]](#)
- [Creating User-Provided Service Instances \[page 252\]](#)
- [Deleting Service Instances \[page 256\]](#)
- [Updating Service Instances \[page 258\]](#)

## SAP Service Manager

You can also access services for the Cloud Foundry environment through SAP Service Manager, the central registry for service brokers and platforms. It's tightly integrated with SAP BTP and allows you to consume services in connected runtime environments and to manage platforms, service brokers, service instances, and service bindings.

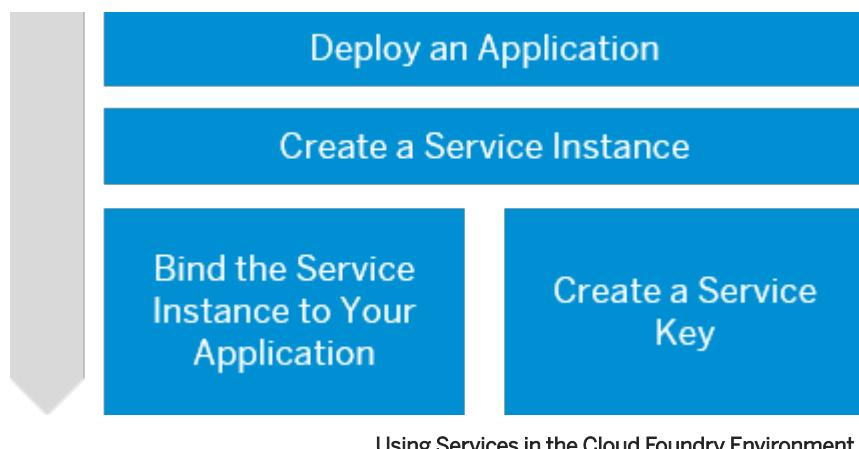
- For more information, see [SAP Service Manager](#).
- To find out if SAP Service Manager is available in your desired region, search for it in the [SAP Discovery Center - Service Catalog](#). In the *Pricing* tab, choose the globe icon to display a map of available regions for this service.

### 4.1.3.4.1 About Services

In the Cloud Foundry environment, you usually enable services by creating a service instance using either the SAP BTP cockpit or the Cloud Foundry command line interface (cf CLI), and binding that instance to your application.

In a PaaS environment, all external dependencies, such as databases, messaging systems, files systems, and so on, are services. In the Cloud Foundry environment, services are offered in a marketplace, from which users can create service instances on-demand. A service instances is a single instantiation of a service running on SAP BTP. Service instances are created using a specific service plan. A service plan is a configuration variant of a service. For example, a database may be configured with various "t-shirt sizes", each of which is a different service plan.

To integrate services with applications, the service credentials must be delivered to the application. To do so, you can bind service instances to your application to automatically deliver these credentials to your application. Or you can use service keys to generate credentials to communicate directly with a service instance. As shown in the figure below, you can deploy an application first and then bind it to a service instance:



Alternatively, you can also bind the service instance to your application as part of the application push via the application manifest. For more information, see <https://docs.cloudfoundry.org/devguide/deploy-apps/manifest.html#services-block>.

### Note

Have in mind that you need to create a service instance first before you integrate it into your application manifest.

The Cloud Foundry environment also allows you to work with user-provided service instances. User-provided service instances enable developers to use services that are not available in the marketplace with their applications running in the Cloud Foundry environment. Once created, user-provided service instances behave in the same manner as service instances created through the marketplace. For more information, see [Creating User-Provided Service Instances \[page 252\]](#).

For more conceptual information about using services in the Cloud Foundry environment, see <https://docs.cloudfoundry.org/devguide/services/> .

## Related Information

[Creating Service Instances \[page 250\]](#)

[Binding Service Instances to Applications \[page 253\]](#)

[Creating Service Keys \[page 255\]](#)

[Creating User-Provided Service Instances \[page 252\]](#)

[Solutions and Services \[page 16\]](#)

### 4.1.3.4.2 Creating Service Instances

Use the SAP BTP cockpit or the Cloud Foundry Command Line Interface to create service instances:

- [Create Service Instances Using the Cockpit \[page 250\]](#)
- [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 251\]](#)

You can also create service instances by declaring them as part of your multitarget application (MTA). To learn how to do that, have a look at the service creation parameters.

- [Service Creation Parameters \[page 641\]](#)

#### 4.1.3.4.2.1 Create Service Instances Using the Cockpit

### Note

The content of this page has moved to a new location.

To learn how to create service instances using the cockpit, see

[Creating Service Instances](#).

## 4.1.3.4.2.2 Create Service Instances Using the Cloud Foundry Command Line Interface

You can use the Cloud Foundry Command Line Interface (cf CLI) to create service instances.

### Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry.  
For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- If you are working in an enterprise account, you need to add quotas to the services you purchased in your subaccount before they appear in the service marketplace. Otherwise, only default free-of-charge services are listed. Quotas are automatically assigned to the resources available in trial accounts.  
For more information, see [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#).

### Procedure

1. (Optional) Open a command line and enter the following string to list all services and service plans that are available in your org:

```
cf marketplace
```

2. Run the following command to create a service instance:

```
cf create-service SERVICE PLAN SERVICE_INSTANCE
```

Specify the following parameters:

- **SERVICE**: The name of the service you want to create an instance of.
- **PLAN**: The name of the service plan you want to use.
- **SERVICE\_INSTANCE**: The new name for your service instance. Use only alphanumeric characters, hyphens, and underscores.

### Related Information

[Binding Service Instances to Applications \[page 253\]](#)

[Creating User-Provided Service Instances \[page 252\]](#)

[About Services \[page 249\]](#)

#### 4.1.3.4.3 Creating User-Provided Service Instances

User-provided service instances enable you to use services that are not available in the marketplace with your applications running in the Cloud Foundry environment.

You can create user-provided service instances using the SAP BTP cockpit or the Cloud Foundry Command Line Interface:

- [Create User-Provided Service Instances Using the Cockpit \[page 252\]](#)
- [Create User-Provided Service Instances Using the Cloud Foundry Command Line Interface \[page 252\]](#)

For more information on user-provided service instances, see <https://docs.cloudfoundry.org/devguide/services/user-provided.html>.

##### 4.1.3.4.3.1 Create User-Provided Service Instances Using the Cockpit

###### Note

The content of this page has moved to a new location.

To learn how to create user-provided service instances using the cockpit, see [Creating User-Provided Service Instances in Cloud Foundry](#).

##### 4.1.3.4.3.2 Create User-Provided Service Instances Using the Cloud Foundry Command Line Interface

Use the Cloud Foundry Command Line Interface to make a user-provided service instance available to Cloud Foundry applications.

#### Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- Obtain a URL, port, user name, and password for communicating with a service that is not available in the marketplace.

## Context

For more information on user-provided service instances, see <https://docs.cloudfoundry.org/devguide/services/user-provided.html>.

## Procedure

Open a command line and enter the following string to create a user-provided service instance:

```
cf create-user-provided-service SERVICE_INSTANCE [-p CREDENTIALS]
```

Specify the following parameters:

- **SERVICE\_INSTANCE**: The new name for your service instance.
- **CREDENTIALS**: Credentials as JSON

## Next Steps

To bind your application to the user-provided service instance, follow the steps described in [Bind Service Instances to Applications Using the Cloud Foundry Command Line Interface \[page 254\]](#).

## Related Information

[Creating Service Instances \[page 250\]](#)

[Creating Service Keys \[page 255\]](#)

[About Services \[page 249\]](#)

### 4.1.3.4.4 Binding Service Instances to Applications

Use the SAP BTP cockpit or the Cloud Foundry Command Line Interface to bind service instances to applications:

- [Bind Service Instances to Applications Using the Cockpit \[page 254\]](#)
- [Bind Service Instances to Applications Using the Cloud Foundry Command Line Interface \[page 254\]](#)

You can also bind service instances by declaring them as part of your multitarget application (MTA). To learn how to do that, have a look at the service binding parameters.

- [Service Binding Parameters \[page 643\]](#)

#### 4.1.3.4.4.1 Bind Service Instances to Applications Using the Cockpit

##### Note

The content of this page has moved to a new location.

To learn how to bind service instances to applications using the cockpit, see [Binding Service Instances to Cloud Foundry Applications](#).

#### 4.1.3.4.4.2 Bind Service Instances to Applications Using the Cloud Foundry Command Line Interface

You can bind service instances to applications using the Cloud Foundry Command Line Interface (cf CLI).

#### Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- Deploy an application in the same space in which you plan to create the service instance. For more information, see [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#).
- Create a service instance. For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 251\]](#).

#### Procedure

Open a command line and enter the following string:

```
cf bind-service APP-NAME SERVICE_INSTANCE [-c PARAMETERS_AS_JSON]
```

Specify the following parameters:

- APP\_NAME: Name of the application.
- SERVICE\_INSTANCE: Name of the service instance.
- -c: (Optional) Provide service-specific configuration parameters either in a valid JSON object in-line or in a parameters file. This file reference can be an absolute or relative path to a file.

## Related Information

[Creating Service Instances \[page 250\]](#)

[Creating Service Keys \[page 255\]](#)

[About Services \[page 249\]](#)

### 4.1.3.4.5 Creating Service Keys

You can use service keys to generate credentials to communicate directly with a service instance. Once you configure them for your service, local clients, apps in other spaces, or entities outside your deployment can access your service with these keys.

You can use the SAP BTP cockpit or the Cloud Foundry Command Line Interface to create service keys:

- [Creating Service Keys Using the Cockpit](#)
- [Create Service Keys Using the Cloud Foundry Command Line Interface \[page 255\]](#)

For more information on service keys, see <https://docs.cloudfoundry.org/devguide/services/service-keys.html>.

#### 4.1.3.4.5.1 Create Service Keys Using the Cockpit

##### ⓘ Note

The content of this page has moved to a new location.

To learn how to create service keys using the cockpit, see [Creating Service Keys in Cloud Foundry](#).

#### 4.1.3.4.5.2 Create Service Keys Using the Cloud Foundry Command Line Interface

Use the Cloud Foundry Command Line Interface to create a service key.

## Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- Create a service instance. For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 251\]](#).

## Procedure

Run the following command to create a service key:

```
cf create-service-key SERVICE_INSTANCE SERVICE_KEY [-c PARAMETERS_AS_JSON]
```

Specify the following parameters:

- **SERVICE\_INSTANCE**: Name of the service instance.
- **SERVICE\_KEY**: Name for the service key.
- **-c**: (Optional) Provide service-specific configuration parameters either in a valid JSON object in-line or in a parameters file. This file reference can be an absolute or relative path to a file.

## Results

Local clients, apps in other spaces, or entities outside your deployment can now access your service instance with this key.

## Related Information

[Creating Service Instances \[page 250\]](#)

[Binding Service Instances to Applications \[page 253\]](#)

[About Services \[page 249\]](#)

### 4.1.3.4.6 Deleting Service Instances

Use the SAP BTP cockpit or the Cloud Foundry Command Line Interface to delete service instances:

#### ⚠ Caution

Be aware that instances will be deleted ultimately. There is no way to revoke this step.

- [Delete Service Instances Using the Cockpit \[page 257\]](#)
- [Delete Service Instances Using the Cloud Foundry Command Line Interface \[page 257\]](#)

You can also delete service instances using the Multitarget Application plug-in for the Cloud Foundry command line interface. This works with the command `deploy`, `bg-deploy`, and `undeploy` with specifying the `--delete-services` option. To learn how to do that, have a look at the multitarget application commands.

- [Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#)

## 4.1.3.4.6.1 Delete Service Instances Using the Cockpit

### ⓘ Note

The content of this page has moved to a new location.

To learn how to delete service instances using the cockpit, see [Deleting Service Instances](#).

## 4.1.3.4.6.2 Delete Service Instances Using the Cloud Foundry Command Line Interface

You can use the Cloud Foundry Command Line Interface (cf CLI) to delete service instances.

### Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry.  
For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

### Procedure

1. Open a command line and log in.

```
cf login -a <API endpoint>
```

2. (Optional) List all services and bound apps in your org:

```
cf services
```

3. Unbind the service from any application.

```
cf unbind-service APP_NAME SERVICE_INSTANCE
```

4. (Optional) List all service keys for your service instance.

```
cf service-keys SERVICE_INSTANCE
```

5. Delete any service key from the service instance.

```
cf delete-service-key SERVICE_INSTANCE SERVICE_KEY
```

6. Run the following command to delete a service instance:

```
cf delete-service SERVICE_INSTANCE
```

Specify the following parameters:

- **SERVICE\_INSTANCE**: The name of your service instance.
- **APP\_NAME**: The name of an application your service instance is connected to.
- **SERVICE\_KEY**: The name of the service key of your service instance.

## Related Information

[About Services \[page 249\]](#)

[Deleting Service Instances \[page 256\]](#)

### 4.1.3.4.7 Updating Service Instances

Use the Cloud Foundry Command Line Interface to update service instances:

- [Update Service Instances Using the SAP BTP Cockpit or Cloud Foundry Command Line Interface \[page 258\]](#)

You can also update service instances inside a multitarget application if the service broker supports updates. Change the deployment descriptor file or a configuration file for a service instance inside your multitarget application and deploy your application to trigger an update of the respective service instance.

#### 4.1.3.4.7.1 Update Service Instances Using the SAP BTP Cockpit or Cloud Foundry Command Line Interface

##### Note

We have redesigned SAP BTP cockpit screens to view and manage service instances.

For more information about the procedure to update service instances using the cockpit, see [Updating Service Instances](#).

To learn how to update service instances using the Cloud Foundry Command Line Interface, continue reading this document.

Use Cloud Foundry Command Line Interface to update service instances:

## Prerequisites

- Download and install the cf CLI and log on to Cloud Foundry.

For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

## Context

You are using a service instance, for which you want to change the plan or the service-specific configuration.

## Procedure

1. (Optional) Open a command line and enter the following string to list all services in your space:

```
cf services
```

2. (Optional) Enter the following string to list the details of your service:

```
cf service SERVICE_INSTANCE
```

3. Run the following command to update your service instance:

```
cf update-service SERVICE_INSTANCE [-p NEW_PLAN] [-c PARAMETERS_AS_JSON] [-t TAGS]
```

Specify the following parameters:

- **SERVICE\_INSTANCE**: The name of your service instance as shown when executing `cf services`.

### 4.1.3.4.8 Boosters

Boosters are a set of guided interactive steps that enable you to select, configure, and consume services on SAP BTP to achieve a specific technical goal.

You can access them directly from your desired global account in the SAP BTP cockpit, by choosing **Boosters** in the navigation menu. This leads to a page where you can find an overview of all available boosters, grouped by capability. From this overview page, you can get quick information about a booster, start a booster, or choose a tile to access the booster details.

## Booster Details

The details of boosters are organized in 3 tabs:

- [Overview](#)

Here, you can get information about what the booster does, its key features and how that particular booster can help you.

- [Components](#)  
Here, you can see all the components that are required for the booster to run.
- [Additional Resources](#)  
Here, you find a list of additional information resources where you can learn more about the concepts and components mentioned in the booster.

Boosters automate processes and always achieve a technical goal, often in the form of an artifact. Artifacts are entities that you develop which may consume technical components (for example, services). Examples of artifacts include applications, content for integration and workflows, or even documents.

When you start a booster, a wizard opens up which guides you through a set of steps. Following these steps enables you to reach the result described in the booster overview.

## Related Information

[Create a Subaccount \[page 2173\]](#)

[Navigate in the Cockpit \[page 2142\]](#)

[Create Spaces \[page 2441\]](#)

### 4.1.3.5 Development Languages

Learn about the development languages supported in the SAP BTP, Cloud Foundry environment.

The polyglot nature of SAP BTP means that you have the freedom to choose between various development languages. While we do recommend using Java and Node.js for most use cases, you're free to use the language that best suits your development needs.

Find more information on the three main languages supported by SAP by following the links below:

- [Developing Java in the Cloud Foundry Environment \[page 261\]](#)
- [Developing Node.js in the Cloud Foundry Environment \[page 343\]](#)
- [Developing Python in the Cloud Foundry Environment \[page 354\]](#)

#### ⓘ Note

Please be informed that the SAP BTP, Cloud Foundry runtime Cloud Service is an open Platform-as-a-Service targeted at microservice development and orchestration. SAP is a member of the Cloud Foundry Foundation, which provides Cloud Foundry, a model for cloud-native application delivery. SAP allows use of certain, publicly available, system buildpacks that are supported by the Cloud Service. However, SAP is not in possession or control of, and cannot be responsible for the content, operation, or use of such system buildpacks, which are not part of the Cloud Service (including SAP Business Technology Platform).

SAP may, without notice, remove any system buildpacks at any time and at SAP's own discretion.

## Tutorials

Tutorial Navigator: [Build an Application in the Cloud Foundry Runtime](#) 

## Guided Answers

Find solutions to some Java, Python and Node.js issues in the troubleshooting section: [SAP Cloud Foundry Buildpacks](#) 

## Related Information

[Best Practices \[page 236\]](#)

### 4.1.3.5.1 Developing Java in the Cloud Foundry Environment

Find selected information for Java development on SAP BTP, Cloud Foundry and references to more detailed sources.

## Usage

To use a buildpack, specify its name when deploying a Java application to SAP BTP, Cloud Foundry. For example:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack_jakarta  
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack  
cf push -f <PATH_TO_APP_MANIFEST> -b java_buildpack
```

You can also use the *buildpack* or *buildpacks* attribute to specify it in the `manifest.yml` file:

```
---  
applications:  
- name: <APP_NAME>  
  buildpacks:  
    - sap_java_buildpack_jakarta  
    ...
```

or in the `mtad.yml` file of your archive:

```
...
modules:
  - name: <APP_NAME>
    type: java.tomcat
    path: <path_to_archive>
    properties:
      ...
    parameters:
      ...
    memory: 512M
    buildpack: sap_java_buildpack_jakarta
...
...
```

## What's New

To check the latest news and updates about SAP Java Buildpack, go to its release notes: [What's New for SAP Java Buildpack](#)

## Troubleshooting

If you encounter an issue while using SAP Java Buildpack, you can create an incident for your specific problem, using support component **BC-CP-CF-BLDP**.

## Related Information

[Buildpacks \[page 262\]](#)

[Customizing the JVM Settings \[page 284\]](#)

[Logging and Tracing \[page 311\]](#)

[Debugging Java Applications \[page 316\]](#)

[Bill of Materials \(BOM\) \[page 342\]](#)

## 4.1.3.5.1.1 Buildpacks

You can choose to deploy applications by using the following Java buildpacks:

- [SAP Java Buildpack 1 \[page 263\]](#) – an SAP-managed buildpack that supports Java 8, 11, 17, TomEE 7 and Tomcat 9
- [SAP Java Buildpack 2 \[page 267\]](#) – an SAP-managed buildpack that supports Java 17, 21 and Tomcat 10
- [Community Java Buildpack \[page 270\]](#) – an open-source product maintained by the [java-buildpack](#) community

## 4.1.3.5.1.1.1 SAP Java Buildpack 1

SAP Java Buildpack 1 is a Cloud Foundry buildpack for running JVM-based applications.

This buildpack supports Java 8, 11, and 17, as well as the following runtimes:

- [TomEE 7 \[page 272\]](#)
- [Tomcat 9 \[page 275\]](#)
- [Java Main \[page 281\]](#)

### ⚠ Caution

Bear in mind that SAP Java Buildpack 1 has been deprecated and is going to be **removed** from SAP BTP, Cloud Foundry environment on **July 1, 2025!**

For more information, see: [Release Notes - August 22](#)

## Usage

To use this buildpack, specify its name when deploying a Java application to the SAP BTP, Cloud Foundry environment:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack
```

You can also use the buildpack attribute to specify it in the `manifest.yml` file:

```
---
applications:
- name: <APP_NAME>
  buildpacks:
    - sap_java_buildpack
  ...
  ...
```

or in the `mtad.yml` file of your archive:

```
...
modules:
  - name: <APP_NAME>
    type: java.tomcat
    path: <path_to_archive>
    properties:
      ...
parameters:
  ...
  memory: 512M
  buildpack: sap_java_buildpack
  ...
  ...
```

## Buildpack Versions

The SAP BTP, Cloud Foundry environment provides four versions of SAP Java Buildpack 1 as part of its system buildpacks:

- *sap\_java\_buildpack* – Holds the latest available version of SAP Java Buildpack 1. All new features and fixes are provided with this version.
- *sap\_java\_buildpack\_<version\_latest>* – Holds the latest available version of SAP Java Buildpack 1. It's available for a limited timeframe (4 to 6 weeks).
- *sap\_java\_buildpack\_<version\_previous>* – This version used to be latest in the previous update of the SAP BTP, Cloud Foundry environment. It's available for a limited timeframe (4 to 6 weeks).
- *sap\_java\_buildpack\_<version\_before\_previous>* – This version used to be latest before two updates of the SAP BTP, Cloud Foundry environment. It's available for a limited timeframe (4 to 6 weeks).

To check these versions, proceed as follows:

1. Log in to a particular SAP BTP region and subaccount. For example, if your region is **eu10**, run:

```
cf api https://api.cf.eu10.hana.ondemand.com
```

2. Then run:

```
cf buildpacks
```

### How to use versions of SAP Java Buildpack 1?

- **Option 1:** Use the default one – *sap\_java\_buildpack*

You take advantage of all latest features and fixes in SAP Java Buildpack 1. This way, it's guaranteed that the buildpack is always available. The drawback in this case is the limited time for adoption, if it's needed. In such a scenario, applications can fall back to an older version temporarily to avoid any downtime.

- **Option 2:** Set a particular version – *sap\_java\_buildpack\_<version\_suffix>*

Bear in mind that this version will only exist for a limited amount of time. This may lead to the situation where application restage is failing because the used version of the buildpack is no longer available. To avoid this, we recommend that you follow the updates of the buildpack and test your applications with its latest version. Developers should never allow their applications to run on an outdated buildpack version.

#### Example:

Let's say that the latest version of SAP Java Buildpack 1 is **1.80.0**. Then, the output of the `cf buildpacks` command would be:

```
buildpack          position  enabled  locked  filename
sap_java_buildpack    1        true     false
sap_java_buildpack-1.80.0.zip
sap_java_buildpack_1_80   2        true     false
sap_java_buildpack-1.80.0.zip
sap_java_buildpack_1_79   3        true     false
sap_java_buildpack-1.79.0.zip
sap_java_buildpack_1_78   4        true     false
sap_java_buildpack-1.78.0.zip
```

When SAP Java Buildpack 1 is updated on the SAP BTP, Cloud Foundry environment from version **1.80.0** to **1.81.0**, the list will change to:

```
- You take advantage of all latest features and fixes in SAP
JavaBuildpack      position  enabled  locked  filename
```

sap_java_buildpack	1	true	false
sap_java_buildpack-1.81.0.zip	2	true	false
sap_java_buildpack_1_81	3	true	false
sap_java_buildpack-1.81.0.zip	4	true	false
sap_java_buildpack_1_80	5	true	false
sap_java_buildpack-1.80.0.zip	6	true	false
sap_java_buildpack_1_79	7	true	false
sap_java_buildpack-1.79.0.zip	8	true	false

This means that `sap_java_buildpack_1_78` will no longer be available for applications.

### Note

No fixes will be provided to older versions of the buildpack. Fixes, including security ones, will be part of the latest version.

## How to switch to a specific buildpack version?

To use `sap_java_buildpack_<version_suffix>`, specify its name when pushing an application to SAP BTP, Cloud Foundry environment:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack_<version_suffix>
```

Alternatively, you can specify the buildpack in the `manifest.yml` file.

For example:

```
---
applications:
- name: myapp
  memory: 128M
  path: ./target/myapp.war
  instances: 1
  buildpacks:
    - sap_java_buildpack_1_80
```

You can do the same in the `mtad.yml` of your `mtar` archive:

```
...
modules:
  - name: myapp
    type: java.tomcat
    path: ./target/myapp.war
    properties:
      ...
    parameters:
      ...
      memory: 512M
      buildpack: sap_java_buildpack_1_80
...
```

## Supported Java Versions

SAP Java Buildpack 1 (`sap_java_buildpack`) supports the following Java versions:

- Java **8** – default version when you use SAP JVM ([it provides a JRE and JDK with Java 8](#))
- Java **11** – default version when you use SapMachine ([it provides a JRE with Java 11](#))

- Java 17 – possible version when you use SapMachine (*it provides a JRE with Java 17*)

To learn how to configure your application to use SapMachine JRE and JDK, see: [SapMachine \[page 290\]](#)

## Components

SAP Java Buildpack 1 provides the following components in the application container (<APP\_ROOT\_DIR>/app/META-INF/.sap\_java\_buildpack):

- Runtimes – [Tomcat 9 \[page 275\]](#), [TomEE 7 \[page 272\]](#), and [Java Main \[page 281\]](#)
- [SapMachine \[page 290\]](#)
- [Memory Calculator V2 \[page 295\]](#) - default
- [Memory Calculator V1 \(SAP JVM Memory Calculator\) \[page 299\]](#) - optional
- [SAP BTP Security Services Integration Libraries](#) – version 2.x

To check all the components regularly updated in the SAP Java Buildpack 1.x releases, see: [SAP Java Buildpack BOM](#)

## Async Servlets

In the current version of the SAP Java Buildpack 1, the async servlets are *not supported*.

**Reason:** Some of the [values](#) that SAP Java Buildpack 1 brings to Tomcat and TomEE 7 are not "async enabled".

## What's New

To see the latest news and updates about SAP Java Buildpack 1, regularly check the release notes on the [What's New portal](#).

## Troubleshooting

If you encounter an issue while using SAP Java Buildpack 1, you can:

- Search for your problem in our Guided Answers: [SAP Java Buildpack](#)
- Create an incident for your specific problem, using support component **BC-CP-CF-BLDP**. To provide the necessary details, use the following template: [Initial Problem-Related Data](#)

See also:

- SAP Note: [3261748](#) SapMachine is replacing OpenJDK 11 & 17 in java\_buildpack
- SAP Note: [3214025](#) Migrating Java Applications from TomEE 1.7 to TomEE 7

## Java Tutorial

The following tutorial will guide you through creating a Java application in Cloud Foundry Command Line Interface (cf CLI), consuming Cloud Foundry services, and setting up authentication and authorization checks. See: [Create an Application with SAP Java Buildpack](#)

### 4.1.3.5.1.1.2 SAP Java Buildpack 2

SAP Java Buildpack 2 is a Cloud Foundry buildpack for running SapMachine-based applications.

This buildpack supports Java 17 and 21, as well as the following runtimes:

- [Tomcat 10 \[page 278\]](#)
- [Java Main \[page 281\]](#)

## Usage

To use this buildpack, specify its name when deploying a Java (Jakarta) application to the SAP BTP, Cloud Foundry environment:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack_jakarta
```

You can also use the buildpack attribute to specify it in the `manifest.yml` file:

```
---
applications:
- name: <APP_NAME>
  buildpacks:
    - sap_java_buildpack_jakarta
  ...
  
```

or in the `mtad.yml` file of your archive:

```
...
modules:
- name: <APP_NAME>
  type: java.tomcat
  path: <path_to_archive>
  properties:
  ...
parameters:
  ...
  memory: 512M
  buildpack: sap_java_buildpack_jakarta
  ...
  
```

## Buildpack Versioning

The SAP BTP, Cloud Foundry environment provides four versions of SAP Java Buildpack 2 as part of its system buildpacks:

- [\*sap\\_java\\_buildpack\\_jakarta\*](#) – Holds the latest available version of SAP Java Buildpack 2. All new features and fixes are provided with this version.
- [\*sap\\_java\\_buildpack\\_jakarta\\_<version\\_latest>\*](#) – Holds the latest available version of SAP Java Buildpack 2. It's available for a limited timeframe (4 to 6 weeks).
- [\*sap\\_java\\_buildpack\\_jakarta\\_<version\\_previous>\*](#) – This version used to be latest in the previous update of the SAP BTP, Cloud Foundry environment. It's available for a limited timeframe (4 to 6 weeks).
- [\*sap\\_java\\_buildpack\\_jakarta\\_<version\\_before\\_previous>\*](#) – This version used to be latest before two updates of the SAP BTP, Cloud Foundry environment. It's available for a limited timeframe (4 to 6 weeks).

To check these versions, proceed as follows:

1. Log in to a particular SAP BTP region and subaccount. For example, if your region is **eu10**, run:

```
cf api https://api.cf.eu10.hana.ondemand.com
```

2. Then run:

```
cf buildpacks
```

## How to use versions of SAP Java Buildpack 2?

- **Option 1:** Use the default one – [\*sap\\_java\\_buildpack\\_jakarta\*](#)

You take advantage of all latest features and fixes in SAP Java Buildpack 2. This way, it's guaranteed that the buildpack is always available. The drawback in this case is the limited time for adoption, if it's needed. In such a scenario, applications can fall back to an older version temporarily to avoid any downtime.

- **Option 2:** Set a particular version – [\*sap\\_java\\_buildpack\\_jakarta\\_<version\\_suffix>\*](#)

Bear in mind that this version will only exist for a limited amount of time. This may lead to the situation where application restage is failing because the used version of the buildpack is no longer available. To avoid this, we recommend that you follow the updates of the buildpack and test your applications with its latest version. Developers should never allow their applications to run on an outdated buildpack version.

### Example:

Let's say that the latest version of SAP Java Buildpack 2 is **2.10.0**. Then, the output of the `cf buildpacks` command would be:

```
buildpack          position  enabled  locked  filename
sap_java_buildpack_jakarta           1        true    false
sap_java_buildpack_jakarta-2.10.0.zip
sap_java_buildpack_jakarta_2_10       2        true    false
sap_java_buildpack_jakarta-2.10.0.zip
sap_java_buildpack_jakarta_2_9        3        true    false
sap_java_buildpack_jakarta-2.9.0.zip
sap_java_buildpack_jakarta_2_8        4        true    false
sap_java_buildpack_jakarta-2.8.0.zip
```

When SAP Java Buildpack 2 is updated on the SAP BTP, Cloud Foundry environment from version **2.10.0** to **2.11.0**, the list will change to:

```
buildpack          position  enabled  locked  filename
```

sap_java_buildpack_jakarta	1	true	false
sap_java_buildpack_jakarta-2.11.0.zip	2	true	false
sap_java_buildpack_jakarta_2_11	2	true	false
sap_java_buildpack_jakarta-2.11.0.zip	3	true	false
sap_java_buildpack_jakarta_2_10	3	true	false
sap_java_buildpack_jakarta-2.10.0.zip	4	true	false
sap_java_buildpack_jakarta_2_9	4	true	false
sap_java_buildpack_jakarta-2.9.0.zip			

This means that `sap_java_buildpack_jakarta_2_8` will no longer be available for applications.

### Note

No fixes will be provided to older versions of the buildpack. Fixes, including security ones, will be part of the latest version.

## How to switch to a specific buildpack version?

To use `sap_java_buildpack_jakarta_<version_suffix>`, specify its name when pushing an application to the SAP BTP, Cloud Foundry environment:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack_jakarta_<version_suffix>
```

Alternatively, you can specify the buildpack in the `manifest.yml` file.

For example:

```
---
applications:
- name: myapp
  memory: 128M
  path: ./target/myapp.war
  instances: 1
  buildpacks:
    - sap_java_buildpack_jakarta_2_9
```

You can do the same in the `mtad.yml` of your `mtar` archive:

```
...
modules:
  - name: myapp
    type: java.tomcat
    path: ./target/myapp.war
    properties:
      ...
parameters:
  ...
  memory: 512M
  buildpack: sap_java_buildpack_jakarta_2_9
...
```

## Supported Java Versions

SAP Java Buildpack 2 (`sap_java_buildpack_jakarta`) supports the following Java versions:

- Java **17** – default version. You can obtain it by using SapMachine 17 (*it provides a JRE with Java 17*)
- Java **21** – you can obtain it by using SapMachine 21 (*it provides a JRE with Java 21*)

To learn how to configure your application to use SapMachine JRE and JDK, see: [SapMachine \[page 290\]](#)

## Components

SAP Java Buildpack 2 provides the following components in the application container (<APP\_ROOT\_DIR>/app/META-INF/.sap\_java\_buildpack\_jakarta):

- Runtimes – [Tomcat 10 \[page 278\]](#) and [Java Main \[page 281\]](#)
- [SapMachine \[page 290\]](#)
- [Memory Calculator V2 \[page 295\]](#)
- [SAP BTP Security Services Integration Libraries](#) – version 3.x

To check all the components regularly updated in the SAP Java Buildpack 2.x releases, see: [SAP Java Buildpack BOM](#)

## What's New

To see the latest news and updates about SAP Java Buildpack 2, regularly check the release notes on the [What's New portal](#).

## Troubleshooting

If you encounter an issue while using SAP Java Buildpack 2, you can:

- Search for your problem in our Guided Answers: [SAP Java Buildpack](#)
- Create an incident for your specific problem, using support component **BC-CP-CF-BLDP**. To provide the necessary details, use the following template: [Initial Problem-Related Data](#)

## 4.1.3.5.1.1.3 Community Java Buildpack

Find SAP-managed Cloud Foundry Java components and updates that belong to the community [java-buildpack](#).

## Buildpack Versioning

The SAP BTP, Cloud Foundry environment provides one recent version of `java_buildpack` as part of its system buildpacks. To check this version, proceed as follows:

1. Log in to a particular SAP BTP region and subaccount. For example, if your region is **eu10**, run:

```
cf api https://api.cf.eu10.hana.ondemand.com
```

2. Then run:

```
cf buildpacks
```

To learn about changes in the Java buildpack's versions and features, regularly check the latest [buildpack releases](#) in the GitHub community page.

## Updates

- **SapMachine** – as of [v4.55](#) of the `java-buildpack`, SAP provides SapMachine 11 and 17 instead of OpenJDK 11 and 17 in the offline `java_buildpack`.  
See SAP Note: [3261748](#) *SapMachine is replacing OpenJDK 11 & 17 in java\_buildpack*

### 4.1.3.5.1.2 Runtimes and Containers

Find out which application runtimes and containers you can use, depending on the Java buildpack your application is using.

- [TomEE 7](#) [page 272] – use this runtime only with SAP Java Buildpack 1
- [Tomcat 9](#) [page 275] – use this runtime only with SAP Java Buildpack 1
- [Tomcat 10](#) [page 278] – use this runtime only with SAP Java Buildpack 2
- [Java Main](#) [page 281] – use the Java Main container with all Java buildpacks

#### ⚠ Restriction

Currently, SAP Java Buildpack 2 does not support Apache TomEE. For this reason, we've temporarily forbidden deploying Java applications running on TomEE. That means, if you set `tomee` as a target runtime, your application will fail during the `staging` phase of the `cf push` command execution.

This restriction will be revoked when we provide a compatible Apache TomEE version for SAP Java Buildpack 2.

## Related Information

## 4.1.3.5.1.2.1 TomEE 7

Web applications deployed with SAP Java Buildpack 1 can run in an TomEE 7 container, which is compatible with Apache Tomcat 8.5 and 9.

Applications can explicitly define the target application container by using the TARGET\_RUNTIME environment variable in the application's `manifest.yml` file

### Example

```
---  
applications:  
- name: myapp  
...  
env:  
  TARGET_RUNTIME: tomee7
```

## Provided APIs

The **tomee7** application runtime container provides the following standard APIs:

Runtime	Full Name	Supported Specifications
tomee7	<b>Apache TomEE 7 (Java EE 7 Web Profile)</b>	<p>Java Servlet 3.1 Java Server Pages (JSP) 2.3 Java Expression Language (EL) 3.0 Java WebSocket 1.1 Java Standard Tag Library (JSTL) 1.2 Java RESTful Web Services (JAX-RS) 2.0 Java Server Faces (JSF) 2.2 Java Debugging Support for Other Languages 1.0 Java Enterprise Beans (EJB) Lite 3.2 Java Transactions API (JTA) 1.2 Java Persistence API (JPA) 2.1 Java Bean Validation 1.1 Java Managed Beans 1.0 Java Interceptors 1.2 Java Annotations 1.2 Java Dependency Injection 1.0 Java Contexts and Dependency Injection 1.1 For a full list of specification versions, see: <a href="#">Apache TomEE 7</a></p>

## Customizing the SAP Java Buildpack Defaults

SAP Java Buildpack 1 provides some default configurations for the Apache TomEE 7 application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature.

Below is a list of all the placeholders that can be customized by the application, along with their default values:

Placeholder	Description	Default Value
connector.maxHttpHeaderSize	The maximum size of the request and response HTTP header, specified in bytes	8192
connector.maxThreads	The maximum number of request processing threads to be created by this Connector, which therefore determines the maximum number of simultaneous requests that can be handled	200
connector.allowTrace	A Boolean value that enables or disables the TRACE HTTP method	false

To configure the HTTP header size, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee7/conf/server.xml':  
  { 'connector.maxHttpHeaderSize':1024 } ]"
```

To configure the maximum number of threads, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee7/conf/server.xml':  
  { 'connector.maxThreads':800 } ]"
```

To enable the TRACE HTTP method, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee7/conf/server.xml':  
  { 'connector.allowTrace':true } ]"
```

## Configure the maximum number of active sessions

SAP Java Buildpack 1 provides the default configurations for unlimited sessions for the Apache TomEE 7 application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature. To limit the number of active sessions, set the `maxActiveSessions` attribute on a `Manager` element. For example:

### Example

```
<Context>  
  <Manager maxActiveSessions="500" />  
</Context>
```

## Configure the session timeout value

To set session timeout value of active sessions, set the `<session-config>` tag in the application's `web.xml` file:

### Example

```
<session-config>
    <session-timeout>1</session-timeout>
</session-config>
```

## Configure the context path attribute

The default value of context path in `server.xml` is "" (Empty String). You can override this default value by using `app_context_root` in the application's `manifest.yml` file. For example:

### Example

```
...
env:
  JBP_CONFIG_TOMCAT: "[tomee7:{app_context_root: test_context_path}]"
...
```

## 4.1.3.5.1.2.2 Tomcat 9

By default, web applications pushed with SAP Java Buildpack 1 are running in an Apache Tomcat 9 container.

Applications can explicitly define the target application container by using the `TARGET_RUNTIME` environment variable in the application's `manifest.yml` file.

### Example

```
---
applications:
- name: myapp
...
env:
  TARGET_RUNTIME: tomcat
```

## Provided APIs

The **tomcat** application runtime container provides the following standard APIs:

Runtime	Full Name	Supported Specification Version
tomcat	<b>Apache Tomcat 9.0</b>  <b> ⓘ Note</b> Only relevant for SAP Java Buildpack 1	Java 8 and later  Java Servlets 4.0  Java Server Pages (JSP) 2.3  Expression Language (EL) 3.0  Debugging Support for Other Languages 1.0  Java API for WebSocket 1.1  Java Authentication Service Provider Interface for Containers (JASPIC) 1.1  For more information, see <a href="#">Apache Tomcat: Tomcat Versions</a> ↗

## Customizing the SAP Java Buildpack Defaults

SAP Java Buildpack 1 provides some default configurations for the Apache Tomcat application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature.

Below is a list of all the placeholders than can be customized by the application, along with their default values:

Placeholder	Description	Default Value
connector.maxHttpHeaderSize	The maximum size of the request and response HTTP header, specified in bytes	8192
connector.maxThreads	The maximum number of request processing threads to be created by this Connector, which therefore determines the maximum number of simultaneous requests that can be handled	200
connector.allowTrace	A Boolean value that enables or disables the TRACE HTTP method	false

To configure the HTTP header size, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/server.xml' :  
  { 'connector.maxHttpHeaderSize':1024 } ]"
```

To configure the maximum number of threads, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/server.xml':  
  { 'connector.maxThreads': 800 } ]"
```

To enable the TRACE HTTP method, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/server.xml':  
  { 'connector.allowTrace': true } ]"
```

## Configure the maximum number of active sessions

SAP Java Buildpack 1 provides the default configurations for unlimited sessions for the Apache Tomcat application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature. To limit the number of active sessions, set the *maxActiveSessions* attribute on a *Manager* element. For example:

### Example

```
<Context>  
  <Manager maxActiveSessions="500" />  
</Context>
```

## Configure the session timeout value

To set session timeout value of active sessions, set the *<session-config>* tag in the application's *web.xml* file:

### Example

```
<session-config>  
  <session-timeout>1</session-timeout>  
</session-config>
```

## Configure the context path attribute

The default value of context path in *server.xml* is "" (Empty String). You can override this default value by using *app\_context\_root* in the application's *manifest.yml* file. For example:

### Example

```
...  
env:  
  JBP_CONFIG_TOMCAT: "[ tomcat:{app_context_root: test_context_path} ]"
```

...

## Configure the cookie processor

In Tomcat 8.5.84, a custom cookie processor has been created, based on the RFC 6265 Cookie Processor. If the PROCESS\_COOKIE environment variable is set to **true**, then this new cookie processor will override the default one.

**Reason:** In Tomcat Apache 8.5.84, the date format used with the *expires* attribute of HTTP cookies was corrected to be compliant with RFC 6265. A single space rather than a single dash is now used to separate the day, month, and year components. For more information, see [Apache Tomcat: Tomcat 8 Changelog](#) ↗

The purpose of the new cookie processor is to set the *Cookie Expire* date to format with '-' (*dash*) delimiter instead of '' (*space*) so that no errors would be thrown.

Below is a sample error message thrown when **not** using the new customization:

### ⌚ Example

```
Sample error: "Invalid cookie header: "set-cookie: username=John; Max-Age=21; Expires=Thu, 17 Aug 2023 13:31:55 GMT". Invalid 'expires' attribute: Thu, 17 Aug 2023 13:31:55 GMT "
```

## 4.1.3.5.1.2.3 Tomcat 10

By default, web applications pushed with SAP Java Buildpack 2 are running in an Apache Tomcat 10 container.

Applications can explicitly define the target application container by using the TARGET\_RUNTIME environment variable in the application's `manifest.yml` file.

### ⌚ Example

```
---
applications:
- name: myapp
  ...
  env:
    TARGET_RUNTIME: tomcat
```

## Provided APIs

The **tomcat** application runtime container provides the following standard APIs:

Runtime	Full Name	Supported Specification Version
tomcat	Apache Tomcat 10.1.x	<p>Java 11 and later</p> <p><b>ⓘ Note</b></p> <p>Only relevant for SAP Java Buildpack 2</p> <p>Java Servlets 6.0</p> <p>Java Server Pages (JSP) 3.1</p> <p>Expression Language (EL) 5.0</p> <p>Debugging Support for Other Languages 2.0</p> <p>Java API for WebSocket 2.1</p> <p>Java Authentication Service Provider Interface for Containers (JASPIC) 3.0</p> <p>For more information, see <a href="#">Apache Tomcat: Tomcat Versions</a> ↗</p>

## Customizing the SAP Java Buildpack Defaults

SAP Java Buildpack 2 provides some default configurations for the Apache Tomcat application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature.

Below is a list of all the placeholders than can be customized by the application, along with their default values:

Placeholder	Description	Default Value
connector.maxHttpHeaderSize	The maximum size of the request and response HTTP header, specified in bytes	8192
connector.maxThreads	The maximum number of request processing threads to be created by this Connector, which therefore determines the maximum number of simultaneous requests that can be handled	200
connector.allowTrace	A Boolean value that enables or disables the TRACE HTTP method	false

To configure the HTTP header size, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomcat/conf/server.xml':  
  {'connector.maxHttpHeaderSize':1024}]"
```

To configure the maximum number of threads, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/server.xml':  
  { 'connector.maxThreads': 800 } ]"
```

To enable the TRACE HTTP method, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/server.xml':  
  { 'connector.allowTrace': true } ]"
```

## Configure the maximum number of active sessions

SAP Java Buildpack 2 provides the default configurations for unlimited sessions for the Apache Tomcat application container. They can be customized by the application with the [Resource Configuration \[page 305\]](#) feature. To limit the number of active sessions, set the *maxActiveSessions* attribute on a *Manager* element. For example:

### Example

```
<Context>  
  <Manager maxActiveSessions="500" />  
</Context>
```

## Configure the session timeout value

To set session timeout value of active sessions, set the *<session-config>* tag in the application's *web.xml* file:

### Example

```
<session-config>  
  <session-timeout>1</session-timeout>  
</session-config>
```

## Configure the context path attribute

The default value of context path in *server.xml* is "" (Empty String). You can override this default value by using *app\_context\_root* in the application's *manifest.yml* file. For example:

### Example

```
...  
env:  
  JBP_CONFIG_TOMCAT: "[ tomcat:{app_context_root: test_context_path} ]"
```

...

## Configure the cookie processor

In Tomcat 8.5.84, a custom cookie processor was created, based on the RFC 6265 Cookie Processor. If the PROCESS\_COOKIE environment variable is set to **true**, then this new cookie processor will override the default one.

**Reason:** In Tomcat Apache 8.5.84, the date format used with the *expires* attribute of HTTP cookies was corrected to be compliant with RFC 6265. A single space rather than a single dash is now used to separate the day, month, and year components. For more information, see [Apache Tomcat: Tomcat 8 Changelog](#).

The purpose of the new cookie processor is to set the *Cookie Expire* date to format with '-' (*dash*) delimiter instead of '' (*space*) so that no errors would be thrown.

Below is a sample error message thrown when **not** using the new customization:

### Example

```
Sample error: "Invalid cookie header: "set-cookie: username=John; Max-Age=21; Expires=Thu, 17 Aug 2023 13:31:55 GMT". Invalid 'expires' attribute: Thu, 17 Aug 2023 13:31:55 GMT "
```

## 4.1.3.5.1.2.4 Java Main

You can create a Java application that starts its own runtime. This allows the usage of frameworks and Java runtimes, such as Spring Boot, Jetty, Undertow, or Netty.

### Prerequisites

- You are **not** using the [Resource Configuration \[page 305\]](#) feature of the buildpack.

### Note

The resource configurations needed for the database connection are not applicable for Java Main applications. For more information about database connections, see: [Configuring a Database Connection \[page 326\]](#)

## Context

In this section, applications like this are referred as *Java Main applications*. The application container provided by SAP Java/Jakarta Buildpack for running Java Main applications is referred as *Java Main container*.

## Procedure

1. Make sure your built JAR archive is configured properly.

Regardless of the tool you use to build your Java application, you have to make sure that the following tasks are performed:

- a. You have built a JAR archive.
- b. You have specified a main class in the META-INF/MANIFEST.MF file of the JAR archive.

### ↔ Sample Code

Manifest.MF

```
Manifest-Version: 1.0
Archiver-Version: Plexus Archiver
Built-By: p1234567
Created-By: Apache Maven 3.3.3
Build-Jdk: 1.8.0_45
Main-Class: com.company_xs.java.main.Controller
```

- c. You have packaged all your dependent libraries in the JAR file, also known as creating an [uber JAR](#) or a [flat JAR](#).

If you are using Maven as your build tool, you can use the `maven-shade-plugin` to perform the above tasks.

### ↔ Sample Code

Sample configuration for the `maven-shade-plugin`

```
<build>
<finalName>java-main-sample</finalName>
<plugins>
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-shade-plugin</artifactId>
    <version>2.4.3</version>
    <configuration>
      <createDependencyReducedPom>false</createDependencyReducedPom>
      <filters>
        <filter>
          <artifact>*:*</artifact>
          <excludes>
            <exclude>META-INF/*.SF</exclude>
            <exclude>META-INF/*.DSA</exclude>
            <exclude>META-INF/*.RSA</exclude>
          </excludes>
        </filter>
      </filters>
    </configuration>
    <executions>
      <execution>
```

```

<phase>package</phase>
<goals>
    <goal>shade</goal>
</goals>
<configuration>
    <transformers>
        <transformer
implementation="org.apache.maven.plugins.shade.resource.ServicesResourceTra
nsformer" />
            <transformer
implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTra
nsformer">
                <manifestEntries>
                    <Main-Class>com.sap.xs.java.main.Controller</Main-Class>
                </manifestEntries>
            </transformer>
        </transformers>
    </configuration>
    </execution>
</executions>
</plugin>
</plugins>
</build>

```

## 2. Configure the manifest.yml.

To be able to push the Java Main application, you need to specify the path to the .jar archive in the manifest.yml file.

### ↔ Sample Code

manifest.yml

```

---
applications:
- name: java-main
  memory: 128M
  path: ./target/java-main-sample.jar
  instances: 1

```

## 3. (Optional) If you use SAP HANA JDBC, we recommend that you include the dependent JAR files in the uber JAR. Then refer these files, as a space separated list, in the class-Path header field of the MANIFEST.MF file. For example:

### ↔ Sample Code

```
Class-Path: jar1-name jar2-name directory-name/jar3-name
```

## 4. Deploy the application on Cloud Foundry. Execute:

```
cf push
```

## Example

To create a Java Main application that is using its own runtime, proceed as follows:

### 1. Create an application, named *sample\_main*. Use Spring Boot for that purpose.

2. Navigate to the `sample_main` directory of the application, using the command line tool, and build it. To do that, execute:

```
mvn clean install
```

3. After the successful build, check that the `sample_main` directory of the application contains a **sample\_main.jar** file.
4. Open the **sample\_main.jar** file and check that the `META-INF/MANIFEST.MF` file contains the `Main-Class` header, whose value is the name of the main class.
5. Add the path to the JAR file in the **manifest.yml** file.

You need to do this to make sure that the application can be pushed to the Cloud Foundry environment. For that purpose, add the following line in the **manifest.yml** file:

```
path: ./target/sample_main.jar
```

6. Finally, deploy the Java Main application. Execute:

```
cf push sample_main
```

### 4.1.3.5.1.3 Customizing the Java Virtual Machine (JVM) Settings

- [Java Options \[page 284\]](#)
- [Java Memory Assistant \[AWS, Azure, or GCP Regions\] \[page 285\]](#)
- [Out-Of-Memory Behavior \[page 286\]](#)
- [Using an Agent \[page 289\]](#)
- [SapMachine \[page 290\]](#)
- [Memory Calculator \[page 294\]](#)

#### 4.1.3.5.1.3.1 Java Options

You can configure the Java properties by defining the `JBP_CONFIG_JAVA_OPTS` environment variable.

Defining the `JBP_CONFIG_JAVA_OPTS` environment variable in the `manifest.yml` file of the application.

##### Sample Code

`manifest.yml`

```
---
applications:
- name: <app-name>
  memory: 512M
...
env:
  JBP_CONFIG_JAVA_OPTS: 'java_opts: ''-DtestJBPCfg=^%PATH^%
-DtestJBPCfg1="test test" -DtestJBPCfg2="%PATH%"'''
```

Defining the JBP\_CONFIG\_JAVA\_OPTS environment variable by using the `cf set-env` command of the Cloud Foundry Command Line Interface (cf CLI).

```
cf set-env myapp JBP_CONFIG_JAVA_OPTS "[java_opts: '-DtestJBPCfg=^%PATH^% -DtestJBPCfg1=\\"test test\\" -DtestJBPCfg2=\\"^%PATH^%\\" ]"
```

## Escaping Strings

### When defining the JBP\_CONFIG\_JAVA\_OPTS in the manifest.yml file

A single quote `'` is used to enclose the JBP\_CONFIG\_JAVA\_OPTS environment variable in the manifest.yml file. Strings containing the following characters must be quoted: `:`, `{`, `}`, `[`, `]`, `,`, `&`, `*`, `#`, `?`, `|`, `-`, `<`, `>`, `=`, `!`, `%`, `@`, `\`.

When a single quote `'` is used, other single quotes `'` in the string must be escaped by doubling it `''`.

### When defining the JBP\_CONFIG\_JAVA\_OPTS by using the cf set-env command

The string must be enclosed in the double quotes `"`. If there are other double quotes in the string, you have escape them using the backslash `\`.

### When defining java options with values containing spaces

In case you need to specify a option value, which has blank spaces in it, you need to surround it with quotes `"`. If the value contains special characters like for example `$`, you have to escape them using the escape character specific for the operating system shell where the application is pushed. In the example below the `$` in the \$PATH string is escaped using the `\` escape character for Linux. Otherwise, if the `$` sign is not escaped, the value of the variable PATH is substituted in the property value.

#### ↳ Sample Code

manifest.yml

```
---
applications:
- name: <app-name>
  memory: 512M
...
env:
  JBP_CONFIG_JAVA_OPTS: 'java_opts: ''-DtestJBPCfg=\$PATH -DtestJBPCfg1="test test" -DtestJBPCfg2="\$PATH"'''
```

## 4.1.3.5.1.3.2 Java Memory Assistant [AWS, Azure, or GCP Regions]

### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Java Memory Assistant is a Java agent that generates heap dumps based on preconfigured conditions in the memory usage of the application. See: [Java Memory Assistant](#)

### ⓘ Note

This public repository went out of maintenance on **Sept 21, 2023**. Nevertheless, its latest version [0.5.0](#) is working fine so you can keep using it.

When enabled in the buildpack, the agent generates two files – **\*.hprof** (heap dump) and **\*.addons**, when the configured memory limits are met.

The **\*.addons** file contains:

- Command-line parameters
- Implemented interfaces for the classes
- Information (name) about transient fields for the classes
- Class and metaspace statistics
- Stack traces of the last OOM errors

See also: [Java Memory Assistant Framework](#)

## 4.1.3.5.1.3.3 Out-Of-Memory Behavior

SAP Java Buildpack prints a histogram of the heap memory to the logs when the JVM encounters a terminal failure.

### ↴ Output Code

```
ERR Stopping VM due to OutOfMemoryError
ERR Resource exhaustion event: the JVM was unable to allocate memory from the
heap.
ERR ResourceExhausted! (1/0)
OUT | Instance Count | Total Bytes | Class
Name
OUT | 30130 | 5556616 |
[C
OUT | 866 | 2485600 |
[B
OUT | 29215 | 701160 | Ljava/lang/
String;
OUT | 3971 | 449528 | Ljava/lang/
Class;
OUT | 9998 | 319936 | Ljava/util/
HashMap$Node;
OUT | 3624 | 318912 | Ljava/lang/reflect/
Method;
OUT | 6429 | 205728 | Ljava/util/concurrent/
ConcurrentHashMap$Node;
OUT | 2821 | 171472 | [Ljava/lang/
Object;
OUT | 40 | 117328 |
[IJ
OUT | 750 | 111096 | [Ljava/util/
HashMap$Node;
OUT | 5618 | 89888 | Ljava/lang/
Object;
```

```

| OUT | 1054 | 74048 | [Ljava/lang/
| String;
| OUT | 696 | 62552 |
| [I
| OUT | 59 | 51920 | [Ljava/util/concurrent/
| ConcurrentHashMap$Node;
| OUT | 1063 | 51024 | Ljava/util/
| HashMap;
| OUT | 854 | 40992 | Lorg/apache/tomcat/util/modeler/
| AttributeInfo;|
...

```

This functionality is **activated** by default.

If you want to deactivate it, add property `-XX:+ExitVMOnOutOfMemoryError` through the `JBP_CONFIG_JAVA_OPTS` property in the application's `manifest.yml` file, like this:

```

---
applications:
- name: <APP_NAME>
  buildpacks:
  - sap_jakarta_buildpack
  env:
    JBP_CONFIG_JAVA_OPTS: 'false, java_opts: ''-XX:+ExitVMOnOutOfMemoryError'''
...

```

## Killing a process with jvmkill

SAP Java Buildpack allows you to set `jvmkill` agent when staging your application. A `jvmkill` agent is added by default to the properties when starting the Java application process. The following configuration is added:

```

-agentpath:METAINF/.sap_java_buildpack/jvm_kill/jvmkill-
<jvmkill_version>.RELEASE-trusty.so=printHeapHistogram=1

```

This is sufficient to print a histogram and kill the process when an out-of-memory (OOM) error occurs.

If you want to disable `jvmkill` and omit its agent being added to the startup properties, use the `SJB_NO_JVMKILL` environment variable, like this:

```

env:
  SJB_NO_JVMKILL: true

```

## Generate Heap Dumps with Java Properties

It's a good practice to generate a `.hprof` heap dump file on the event of `OutOfMemoryError` to analyze and find the root cause of the issue.

Various distributions including [SapMachineJre](#) and [SapMachineJdk](#) provide standard mechanism for generating a heap dump in the event of an OOM error. The following properties can be added on startup for this purpose:

```
-XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=<heap_dump_path>
-XX:OnOutOfMemoryError='kill -9 %p'
```

- **-XX:+HeapDumpOnOutOfMemoryError** triggers heap dump generation under a default path.
- **-XX:HeapDumpPath** specifies *where* to specify the heap dump file.
- **-XX:OnOutOfMemoryError** specifies *what command or script* to be run if OOM occurs. This command usually kills the process.

The heap dump is generated and stored in an application container location. This means, it will be removed once the application instance container is restarted.

#### Note

Killing the process will trigger a restart. Therefore, if you need to investigate further in the heap dump file stored in the application container, the process should not be killed.

## Related Information

[ActiveProcessorCount \[page 288\]](#)

[Cloud Foundry: jvmkill](#) ↗

### 4.1.3.5.1.3.4 ActiveProcessorCount

This JVM option overrides the number of CPUs which the virtual machine uses to calculate the size of thread pools for operations (such as garbage collection).

#### Note

Previously, SAP Java Buildpack used to set JVM option "`-xx:+UseContainerCpuShares`" but it's now replaced by "`-XX:ActiveProcessorCount=<value>`".

The `ActiveProcessorCount` flag prevents out-of-memory (OOM) errors in case a GC algorithm with high internal memory usage has been activated by the JVM in relatively small containers (< 4GB). Larger containers (>4GB) benefit from a higher `ActiveProcessorCount` value.

SAP Java Buildpack automatically sets the JVM option "`-XX:ActiveProcessorCount=<value>`", where `<value>` is defined as follows:

- For container instances < 4GB, the `ActiveProcessorCount` is set to **1**.
- For container instances  $\geq$  4GB, the `ActiveProcessorCount` is set to 1 CPU per 1GB of container memory. For example, if a container instance is 7GB, the value will be set to **7**.

If you want to explicitly set the value of `ActiveProcessorCount`, you can do that through the `JBP_CONFIG_JAVA_OPTS` property in the application's [`manifest.yml`](#) file.

- For SAP Java Buildpack 1:

```
---
applications:
- name: <APP_NAME>
  buildpacks:
  - sap_java_buildpack
  env:
    JBP_CONFIG_JAVA_OPTS: 'java_opts: ''-XX:ActiveProcessorCount=7'''
```

- For SAP Java Buildpack 2:

```
---
applications:
- name: <APP_NAME>
  buildpacks:
  - sap_jakarta_buildpack
  env:
    JBP_CONFIG_JAVA_OPTS: 'java_opts: ''-XX:ActiveProcessorCount=7'''
```

## Related Information

[Out-Of-Memory Behavior \[page 286\]](#)

### 4.1.3.5.1.3.5 Using an Agent

SAP Java Buildpack can work with any Java agent as long as it's included in the .jar or .war archive of your application.

The buildpack extracts the agent when the application is deployed. To check if the agent is extracted to the expected location, run:

```
cf ssh
```

To use an agent with SAP Java Buildpack, set the `<JBP_CONFIG_JAVA_OPTS>` environment variable as follows:

```
env:
  JBP_CONFIG_JAVA_OPTS: 'java_opts: "-javaagent:<PathToYourAgent>"'
```

The Java agent is platform-agnostic, but must be compatible with the version of the JVM you are using (SAP JVM or SapMachine).

## Native Agents

A native agent is a dynamic library. If you want to use one, it must be compatible with the architecture of the platform. For Cloud Foundry, this is Linux x86\_64. As an application developer, make sure that the correct agent is used, and regularly apply updates whenever they're needed.

To use a native agent, set the `<JBP_CONFIG_JAVA_OPTS>` environment variable as follows:

```
env:  
  JBP_CONFIG_JAVA_OPTS: 'java_opts: "-agentpath:<PathToYourAgent>"'
```

### 4.1.3.5.1.3.6 SapMachine

SapMachine is an alternative to SAP JVM, and provides a Java Runtime Environment (JRE) with Java 11, 17, and 21.

SapMachine works with the following application containers:

- [TomEE 7 \[page 272\]](#)
- [Tomcat 9 \[page 275\]](#)
- [Tomcat 10 \[page 278\]](#)
- [Java Main \[page 281\]](#)

#### ⚠ Caution

##### Only relevant to SAP Java Buildpack 1:

Bear in mind that TomEE 7 supports only Java 7 and 8. Thus, even if your TomEE 7 application runs successfully with SapMachine JRE 17, at some point it might crash. Also, TomEE 7 has already [reached end of life](#). See: [Discontinued TomEE versions](#)

## Activation Using JRE

To activate SapMachine **JRE** in SAP Java Buildpack, add the following environment variable:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jre.SAPMachineJRE' ]"  
  ...
```

Without specifying a particular JRE version, your application will use the following default offline JRE, bundled within the buildpack:

- For SAP Java Buildpack 1 – SapMachine JRE 11
- For SAP Java Buildpack 2 – SapMachine JRE 17

To specify a particular JRE version, use environment variable `JBP_CONFIG_SAP_MACHINE_JRE`.

By default, the `use_offline_repository` parameter is set to `true`. That means, your application will use the offline SapMachine JRE provided by the buildpack.

If you set this parameter to `false`, the buildpack will attempt to download SapMachine JRE of a particular version from the GitHub asset repository. This will only work if your Cloud Foundry instance has access to [GitHub: SapMachine](#).

## SapMachine 11

To point to the latest SapMachine JRE 11 version provided by the buildpack, specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{ version: 11.+ }'
```

To make the buildpack download a particular published JRE version (for example, 11.0.13), specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{use_offline_repository: false, version:  
    11.0.13}'  
  ...
```

## SapMachine 17



SAP Java Buildpack 1 and 2 provide a customized SapMachine JRE 17 that contains a [jdk.compiler](#) module.

Use environment variable `JBP_CONFIG_SAP_MACHINE_JRE` in analogical way. For example:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{ version: 17.+ }'
```

To make the buildpack download a particular published JRE version (for example, 17.0.10), specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{use_offline_repository: false, version:  
    17.0.10}'
```

...

## SapMachine 21

### → Tip

SAP Java Buildpack 2 provides a customized SapMachine JRE 21 that contains a [jdk.compiler](#) module.

Use environment variable JBP\_CONFIG\_SAP\_MACHINE\_JRE in analogical way. For example:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jre.SAPMachineJRE' ]"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{ version: 21.+ }'
```

To make the buildpack download a particular published JRE version (for example, 21.0.2), specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jre.SAPMachineJRE' ]"  
    JBP_CONFIG_SAP_MACHINE_JRE: '{use_offline_repository: false, version:  
21.0.2}'  
  ...
```

## Activation Using JDK

To use SapMachine **JDK**, add the following environment variable:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
  ...
```

### ⓘ Note

SapMachine JDK is **not bundled** within the buildpack. Therefore, if you want to use SapMachine JDK 11, 17 or 21, you have to download it from the [GitHub project](#) as an online component.

However, you can use the [jdk.compiler](#) module, which is **provided** by the buildpack as part of the customized SapMachine JRE, v. 17 and 21. To learn more, see section [Activation Using JRE \[page 290\]](#).

To specify an online JDK version (11, 17, or 21), use environment variable JBP\_CONFIG\_SAP\_MACHINE\_JDK.

## SapMachine 11

To make the buildpack download a particular published JDK version (for example, 11.0.22), specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 11.0.22 }'  
  ...
```

To point to the major version of the SapMachine JDK 11, specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 11.+ }'
```

## SapMachine 17

Use environment variable JBP\_CONFIG\_SAP\_MACHINE\_JDK in analogical way. For example:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 17.0.10 }'  
  ...
```

To point to the major version of the SapMachine JDK 17, specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 17.+ }'
```

## SapMachine 21

Use environment variable JBP\_CONFIG\_SAP\_MACHINE\_JDK in analogical way. For example:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"
```

```
JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 21.0.2 }'  
...
```

To point to the major version of the SapMachine JDK 21, specify it the following way:

```
---  
applications:  
- name: <app-name>  
...  
env:  
  JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
  JBP_CONFIG_SAP_MACHINE_JDK: '{ version: 21.+ }'
```

## Related Information

<https://sap.github.io/SapMachine/>

Runtimes and Containers [page 271]

### 4.1.3.5.1.3.7 Memory Calculator

The memory calculator provides a mechanism to fine-tune the Java Virtual Machine (JVM) memory for an application. Its goal is to ensure that applications perform well while not exceeding a container's memory limit.

Below you can find which memory calculators are used in the relevant Java buildpacks.

#### SAP Java Buildpack 1

This buildpack supports two memory calculator versions:

- [Memory Calculator V2 \[page 295\]](#) – activated by default.
- [Memory Calculator V1 \(SAP JVM Memory Calculator\) \[page 299\]](#) – this is an optional memory calculator. You can activate it by adding the following environment variable:

```
---  
applications:  
- name: <app-name>  
...  
env:  
  MEMORY_CALCULATOR_V1: true  
...
```

## SAP Java Buildpack 2

This buildpack supports only [Memory Calculator V2 \[page 295\]](#), which is the same as the one provided by the community Java Buildpack. See: [\(GitHub\) Java Buildpack Memory Calculator](#) ↗

### ⓘ Note

Memory Calculator V1 is deprecated (not available) for SAP Java Buildpack 2. If you try to activate it by setting the MEMORY\_CALCULATOR\_V1 environment variable, an error message will be thrown.

## Community Java Buildpack

This buildpack uses its default and only memory calculator: [\(GitHub\) Java Buildpack Memory Calculator](#) ↗

## Related Information

[Buildpacks \[page 262\]](#)

### 4.1.3.5.1.3.7.1 Memory Calculator V2

Customize the memory options by using the JBP\_CONFIG\_JAVA\_OPTS environment variable:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_JAVA_OPTS: "[ java_opts: '-Xms144M -Xss3M -Xmx444444K  
    -XX:MetaspaceSize=66666K -XX:MaxMetaspaceSize=88888K' ]"
```

## Default Settings

SAP Java Buildpack is delivered with a default built-in configuration of the memory sizing options in YML format - config/sap\_machine\_jre.yml or config/sap\_machine\_jdk.yml (path related to the buildpack archive). These configuration files are parsed during application staging, and the memory configuration specified in them is used for calculating the memory sizes of [stack\\_threads](#), [class\\_count](#) and [headroom](#).

The default structure of the config/sap\_machine\_jre.yml configuration file, for example, is the following:

### ↗ Sample Code

```
config/sap_machine_jre.yml
```

```

# Configuration for JRE repository
---
  repository_root: "https://sap.github.io/SapMachine/assets/cf/jre/{platform}/
{architecture}"
jvmkill_agent:
  version: 1.+
  repository_root: "{default.repository.root}/jvmkill/{platform}/
{architecture}"
memory_calculator:
  version: 3.+
  repository_root: "{default.repository.root}/memory-calculator/{platform}/
{architecture}"
  class_count: 500
  headroom: 7
  stack_threads: 250

```

- `stack_threads` – an estimate of the number of threads that will be used by the application. Default value is **250**.
- `class_count` – an estimate of the number of classes that will be loaded. The default behavior is to estimate this number as a fraction (0.35) of the number of class files in the application.
- `headroom` – percentage of total memory available that is unallocated to cover JVM overhead. Maximum recommended value for headroom is **10**. Default value is **0**.

Depending on which Java version you use (8 or 11), you can customize these three memory options by using the relevant environment variables:

- `JBP_CONFIG_SAPJVM`
- `JBP_CONFIG_SAP_MACHINE_JRE`
- `JBP_CONFIG_SAP_MACHINE_JDK`

## Java 8

### Note

Only relevant for SAP Java Buildpack 1

If you need JRE with Java 8, you have to use SAP JVM. Customize your memory options as follows:

```

---
applications:
- name: <app-name>
  ...
  env:
    JBP_CONFIG_SAPJVM: "[memory_calculator_v2: {stack_threads: 266, class_count:
1001, headroom: 5}]"

```

## Java 11

### ⓘ Note

Only relevant for SAP Java Buildpack 1

If you need JRE with Java 11, you have to use SapMachine 11. Customize your memory options as follows:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jre.SAPMachineJRE' ]"  
    JBP_CONFIG_SAP_MACHINE_JRE: "[version: 11.+ , memory_calculator_v2:  
{stack_threads: 266, class_count: 1001, headroom: 5}]"
```

If you want to point to the SapMachine JDK component, you need to provide a specific version as follows:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 11.0.22 , memory_calculator_v2:  
{stack_threads: 266, class_count: 1001, headroom: 5}]"
```

You can also point to the major version of the SapMachine JDK, in order to always get the latest patch versions. In this case, specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 11.+ , memory_calculator_v2:  
{stack_threads: 266, class_count: 1001, headroom: 5}]"
```

## Java 17

### ⓘ Note

Relevant for all Java buildpacks

If you need JRE with Java 17, you use SapMachine 17. Customize your memory options as follows:

```
---  
applications:  
- name: <app-name>  
  ...
```

```

env:
  JBP_CONFIG_COMPONENTS: "jres:
  ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"
  JBP_CONFIG_SAP_MACHINE_JRE: "[version: 17.+, memory_calculator_v2:
  {stack_threads: 266, class_count: 1001, headroom: 5}]"

```

If you want to point to the SapMachine JDK component, you need to provide a specific version as follows:

```

---
applications:
- name: <app-name>
  ...
  env:
    JBP_CONFIG_COMPONENTS: "jres:
    ['com.sap.xs.java.buildpack.jdk.SAPMachineJDK']"
    JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 17.0.10, memory_calculator_v2:
    {stack_threads: 266, class_count: 1001, headroom: 5}]"

```

You can also point to the major version of the SapMachine JDK, in order to always get the latest patch versions. In this case, specify it the following way:

```

---
applications:
- name: <app-name>
  ...
  env:
    JBP_CONFIG_COMPONENTS: "jres:
    ['com.sap.xs.java.buildpack.jdk.SAPMachineJDK']"
    JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 17.+, memory_calculator_v2:
    {stack_threads: 266, class_count: 1001, headroom: 5}]"

```

## Java 21

### ⓘ Note

Only relevant for SAP Java Buildpack 2

If you need JRE with Java 21, you use SapMachine 21. Customize your memory options as follows:

```

---
applications:
- name: <app-name>
  ...
  env:
    JBP_CONFIG_COMPONENTS: "jres:
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"
    JBP_CONFIG_SAP_MACHINE_JRE: "[version: 21.+, memory_calculator_v2:
    {stack_threads: 266, class_count: 1001, headroom: 5}]"

```

If you want to point to the SapMachine JDK component, you need to provide a specific version as follows:

```

---
applications:
- name: <app-name>
  ...
  env:
    JBP_CONFIG_COMPONENTS: "jres:
    ['com.sap.xs.java.buildpack.jdk.SAPMachineJDK']"

```

```
JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 21.0.2, memory_calculator_v2:  
{stack_threads: 266, class_count: 1001, headroom: 5}]"
```

You can also point to the major version of the SapMachine JDK, in order to always get the latest patch versions. In this case, specify it the following way:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
[ 'com.sap.xs.java.buildpack.jdk.SAPMachineJDK' ]"  
    JBP_CONFIG_SAP_MACHINE_JDK: "[ version: 21.+, memory_calculator_v2:  
{stack_threads: 266, class_count: 1001, headroom: 5}]"
```

## Related Information

[GitHub: Java Buildpack Memory Calculator](#) 

[SapMachine \[page 290\]](#)

### 4.1.3.5.1.3.7.2 Memory Calculator V1 (SAP JVM Memory Calculator)

This memory calculator is optional, and you can activate it by adding the following environment variable:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    MEMORY_CALCULATOR_V1: true  
  ...
```

#### Restriction

Memory Calculator V1 is only available for [SAP Java Buildpack 1 \[page 263\]](#).

If you try to activate it for [SAP Java Buildpack 2 \[page 267\]](#), an error will be thrown.

## General Information

When pushing applications to Cloud Foundry, application developers could specify the memory limit of the application.

The main goal of the SAP JVM Memory Calculator is to provide mechanism to fine tune the Java Virtual Machine (JVM) in terms of restricting the JVM's memory to grow below this memory limit.

There are three memory types, which can be sized - **heap**, **metaspace** and **stack**. For each memory type there is a command-line option, which must be passed to the JVM:

- The initial and maximum size of the **heap** memory is controlled by the `-Xms` and `-Xmx` options, respectively.
- The initial and maximum size of the **metaspace** memory is controlled by the `-XX:MetaspaceSize` and `-XX:MaxMetaspaceSize` options, respectively.
- The size of the **stack** is controlled by the `-Xss` option.

## Default Settings

The SAP Java Buildpack is delivered with a default built-in configuration of the memory sizing options in YML format - `config/sapjvm.yml` (path related to the buildpack archive). That configuration file is parsed during application staging, and the memory configuration specified in it is used for calculating the memory sizes of the **heap**, **metaspace** and **stack**.

The default structure of the `config/sapjvm.yml` configuration file is the following:

### Sample Code

`config/sapjvm.yml`

```
---
repository_root: "{default.repository.root}/sapjvm/{platform}/{architecture}"
version: +
default_keystore_pass: changeit
memory_calculator:
  version: 1.+
repository_root: "{default.repository.root}/memory-calculator/{platform}/
{architecture}"
memory_sizes:
  heap:
    metaspace: 64m..
  stack:
    native:
  memory_heuristics:
    heap: 75
    metaspace: 10
    stack: 5
    native: 10
  memory_initials:
    heap: 100%
    metaspace: 100%
  memory_settings:
    codecache:
    directmemory:
memory_calculator_v2:
  version: 1.+
  repository_root: "{default.repository.root}/memory-calculator-v2/{platform}/
{architecture}"
  class_count:
  stack_threads: 250
```

The **memory\_calculator** section encloses the input data for the memory calculation techniques utilized in determining the JVM memory sizing options.

- **heap** – configure sizing options for the Java heap. Affects JVM options `-Xms` and `-Xmx`

- *metaspace* – configure sizing options for the metaspace. Affects JVM options `-XX:MetaspaceSize` and `-XX:MaxMetaspaceSize`
- *stack* – configure sizing options for the stack. Affects JVM option `-Xss`
- *native* – serves to represent the rest of the memory (different from *heap*, *stack*, *metaspace*) in the calculations performed by the SAP JVM Memory Calculator. No JVM options are affected by this setting.
- *memory\_heuristics* – this section defines the proportions between the memory regions addressed by the memory calculator. The ratios above will result in a *heap* space that is about 7.5 times larger than the *metaspace* and *native*; *stack* will be about half of the *metaspace* size.
- *memory\_sizes* – this section defines sizes of the corresponding memory regions. The size of the memory region could be specified in kilobytes (by using the K symbol), megabytes (by using the M symbol) and gigabytes (by using the G symbol).

Syntax	Range	Value that satisfies the Range
120M	120M	120M
120M..	[120M)	>=120M
120M..150M	[120M, 150M]	>=120M & <=150M

- *memory\_initials* - this section defines initial and maximum values for *heap* and *metaspace*. By default, the initial values for *heap* and *metaspace* are set to 100%. That means, the memory calculation will result in `-Xms=-Xmx` and `-XX:MetaspaceSize=-XX:MaxMetaspaceSize`. If those values are set to 50%, then `-Xmx = 2*-Xms` and `-XX:MaxMetaspaceSize=2*-XX:MetaspaceSize`.
- *memory\_settings* - for details, see [Out-Of-Memory Behavior \[page 286\]](#)
- *memory\_calculator\_v2* - for details, see [Memory Calculator V2 \[page 295\]](#)

## Customizing the Default Settings

There are two ways to customize the default settings - during application staging and during application runtime.

### Customizing the defaults during application staging

- Customize the memory options by using the `JBP_CONFIG_SAPJVM` environment variable

```

---  

applications:  

- name: <app-name>  

  memory: 512M  

...  

  env:  

    JBP_CONFIG_SAPJVM:  "[memory_calculator: {memory_heuristics: {heap: 85,  

  stack: 10}}]"
```

- Customize the memory options by using the `JBP_CONFIG_SAPJVM_MEMORY_*` environment variables

```

---  

applications:  

- name: <app-name>  

  memory: 512M
```

```

...
  env:
    JBP_CONFIG_SAPJVM_MEMORY_SIZES:
      'heap:30m..400m,stack:2m..,metaspace:10m..12m'
    JBP_CONFIG_SAPJVM_MEMORY_WEIGHTS: 'heap:5,stack:1,metaspace:3,native:1'
    JBP_CONFIG_SAPJVM_MEMORY_INITIALS: "heap:50%,metaspace:50%"

```

## Customizing the defaults during application runtime

There are several ways to customize the SAP JVM Memory Calculator's settings during the application runtime. All of them include executing the `set-env` command – either on XSA or on Cloud Foundry. Below are some examples:

- By using the `JBP_CONFIG_SAPJVM` environment variable

```

cf set-env sapjbp-memcalc-sample JBP_CONFIG_SAPJVM "[memory_calculator:
{memory_heuristics: {heap: 85, stack: 10}}]"

```

- Customize the memory sizes by using the `JBP_CONFIG_SAPJVM_MEMORY_SIZES` environment variable

```

cf set-env myapp JBP_CONFIG_SAPJVM_MEMORY_SIZES
'heap:30m..400m,stack:2m..,metaspace:10m..12m'

```

- Customize the memory weights by using the `JBP_CONFIG_SAPJVM_MEMORY_WEIGHTS` environment variable

```

cf set-env myapp JBP_CONFIG_SAPJVM_MEMORY_WEIGHTS
'heap:5,stack:1,metaspace:3,native:1'

```

- Customize the memory initials by using the `JBP_CONFIG_SAPJVM_MEMORY_INITIALS` environment variable

```

cf set-env myapp JBP_CONFIG_SAPJVM_MEMORY_INITIALS "heap:50%,metaspace:50%"

```

## Algorithm

When given certain memory limit, the memory calculator will try to calculate memory sizes for `heap`, `metaspace`, `stack` and `native` in a way that satisfies the proportions configured with the `memory_heuristics`. Then the calculator will validate and adjust those sizes against the configured `memory_sizes`.

Let's say that the calculated value for `heap`, according to the `memory_heuristics`, is 120M. Let's assume that the `memory_sizes` configuration for heap is 10M..100M. Even though 120M goes beyond the configured range, according to `memory_heuristics`, 120M could be allocated for heap and the chosen value for `heap` size would be 100M. The calculation goes on until sizes (`heap`, `metaspace`, `stack`, and `native`) are calculated for all of the regions. Finally, the `memory_initials` will be respected. This would affect the values of `-Xms`, `-Xms`, `-XX:MetaspaceSize`, and `-XX:MaxMetaspaceSize`.

### Example 1 (memory limit of 1G with default settings of the memory calculator)

Memory limit	Memory calculator settings
1G	default

First, the algorithm will try to estimate the number of threads for the given `memory_limit` and `memory_heuristics` given 1M per thread. This way, we'll have `estimated_number_of_threads=((5/100)*1024M) = 51.2M` space for `stack`. Considering 1M per thread (which by default is the `-Xss` size of the SAP JVM), we'll get `estimated_number_of_threads=51.2`

1. Apply the configured `memory_heuristics`:

- `heap`:  $(75/100) * 1024M = 768M$
- `metaspace`:  $(10/100) * 1024M = 104857K$
- `stack`:  $(5/100) * 1024M = 51.2M$
- `native`:  $(10/100) * 1024M = 104858K$

2. Apply the configured `memory_sizes`:

There is a memory range defined for `metaspace`, which is 64.

The value 104857K calculated in step1 for `metaspace` satisfies the range.

3. Apply the `memory_initials`:

- `-Xms`:  $100\% * 768M = 768M$
- `-Xmx`: 768M
- `-XX:MetaspaceSize`:  $100\% * 104857K = 104857K$
- `-XX:MaxMetaspaceSize`: 104857K
- `-Xss`: `stack` / `estimated_number_of_threads` =  $51.2M / 51.2 = 1M$

## Example 2 (memory limit of 1G with customized settings of the memory calculator)

Memory limit	Memory calculator settings
1G	<pre> memory_sizes:   • metaspace: 64m..70m  memory_heuristics:   • heap: 75   • metaspace: 10,   • stack: 5   • native: 10  memory_initials:   • heap: 100%   metaspace: 50% </pre>

First, the algorithm will try to estimate the number of threads for the given `memory_limit` and `memory_heuristics` given 1M per thread. This way, we'll have `estimated_number_of_threads=((5/100)*1024M) = 51.2M` space for `stack`. Considering 1M per thread (which by default is the `-Xss` size of the SAP JVM), we'll get `estimated_number_of_threads=51.2`.

### Round 1:

1. Apply the configured `memory_heuristics`:

- `heap` =  $(75/100) * 1024M = 768M$
- `metaspace` =  $(10/100) * 1024M = 104857K$
- `stack` =  $(5/100) * 1024M = 51.2M$

- *native*:  $(10/100) * 1024M = 104857K$
2. Apply the configured `memory_sizes`:  
The `metaspace` should be between 60m and 70m. Since the value calculated in step 1 goes beyond this range, the chosen value for `metaspace` becomes 70M. This leads to a second round of calculation since there are  $(104857K - 70M) = 33177K$ , which are available to be distributed among the other regions - heap, stack, native.

#### Round 2:

The `metaspace` is already calculated, so its size is subtracted from the memory limit leading to memory available for distribution, which is  $(1G - 70M) = 954M$ .

The same calculation takes place, this time the `metaspace` is not considered because it's already calculated.

1. Apply the `memory_heuristics`:
  - `heap` =  $(75/90) * 954M = 795M$
  - `stack` =  $(5/90) * 954M = 53$
  - `native` =  $(10/90) * 954M = 106M$
2. Apply the `memory_sizes`:  
No settings for `heap`, `stack` or `native`, thus no changes are needed in the calculations from step 1.
3. Apply the `memory_initials`:
  - `-Xms`:  $100\% * 795M = 795M$
  - `-Xmx`:  $795M$
  - `-XX:MetaspaceSize`:  $50\% * 70M = 35M$
  - `-XX:MaxMetaspaceSize`:  $70M$
  - `-Xss`:  $stack / \text{estimated\_number\_of\_threads} = 53 / 51.2 = 1060K$

## Related Information

[Memory Calculator V2 \[page 295\]](#)

### 4.1.3.5.1.4 Overriding Resources in Droplet

Applications can override resources in the droplet by placing directory files that have the same relative path as the droplet root.

The following example shows how an application can override the default Tomcat's `server.xml` file in directory `META-INF/sap_java_buildpack/resources/`.

## Example

1. You have the following file structure in your Java application:

```
META-INF/
```

```
- sap_java_buildpack/
  - resources/
    - tomcat/
      - conf/
        - server.xml/
WEB-INF/
```

- After the application is built and deployed, this will result in the following file structure in the droplet:

```
META-INF/
- .sap_java_buildpack/
  - tomcat
    - conf/
      - server.xml
WEB-INF/
```

- When the Tomcat container is started, the content of the `server.xml` file will come from the application.

This mechanism also allows adding new resources inside the droplet in their respective places, based on where the files are located under `META-INF/sap_java_buildpack/resources/`.

→ Tip

The overriding principle is applicable for both `sap_java_buildpack` and `sap_java_buildpack_jakarta`.

## Related Information

[Resource Configuration \[page 305\]](#)

[Cloud Foundry component glossary](#) ↗

### 4.1.3.5.1.5 Resource Configuration

Both application containers, [Tomcat \[page 275\]](#) and [TomEE 7 \[page 272\]](#), are configured through text configuration files.

For example:

- conf/server.xml
- conf/tomee7.xml
- conf/logging.properties

See: [Apache Tomcat 9 Configuration Reference](#) ↗

Resource configuration feature of SAP Java Buildpack provides means for changing parameterized values in all text files (part of the application container or part of the application files) during staging.

Files can contain multiple key-value pairs containing placeholders, marked as  `${propname}`. To prevent undesired modifications, all files that can be changed must be explicitly listed by the application in a corresponding `resource_configuration.yaml` file, along with the default values for all placeholders.

## Template for `resource_configuration.yml`

```
---
```

```
filepath1:
  key1: value1
  key2: value2
filepath2:
  key1: value1
```

The `filepath1` must start with either **tomcat** or **tomee7** to match the used application container, and then contain a subpath to a resource.

- When changing parametrized values inside application files, the file path should look like this:  
`<application_container>/webapps/ROOT/<path_to_file_relative_to_application_root>`
- When changing parametrized values in the configuration of an application container, the file path should look like this:  
`<application_container>/conf/<path_to_config_file>`

## Example 1

You can create a `resource_configuration.yml` with the following content:

```
tomcat/webapps/ROOT/WEB-INF/mytextfield.txt:
  placeholder: "defaultValue"
tomcat/conf/server.xml:
  connector.maxThreads: 800
```

## Example 2

1. You have a given application with the following application files:

```
src/
  main/
    java/
      resources/
        webapp/
          META-INF/
            sap_java_buildpack/
              config/
                resource_configuration.yml
          WEB-INF/
            mytextfield.txt
```

2. The content of `mytextfield.txt` is:

```
My hometown is ${hometown}.
```

3. The `resource_configuration.yml` looks like this:

```
---
tomcat/webapps/ROOT/WEB-INF/mytextfield.txt:
  hometown: "London"
```

- After the application is staged, the content of `mytextfield.txt` will be:

```
My hometown is London.
```

## Example 3

The following example demonstrates how to provide values for placeholders during staging.

- Use the data from **Example 2** (steps 1-3).
- Then, in the `manifest.yml` file of the application, add the following environment variable:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/webapps/ROOT/WEB-INF/  
  mytextfield.txt': { 'hometown': 'Paris' } ]"
```

This gives the possibility to provide values for placeholders that differ from the default ones provided in the example above.

- After staging of the application, the content of `mytextfield.txt` will be modified to:

```
My hometown is Paris.
```

## Related Information

[Overriding Resources in Droplet \[page 304\]](#)

### 4.1.3.5.1.6 Context Root Redirect

SAP Java Buildpack provides a context root redirect functionality.

When you call a Web application without adding its runtime ([Tomcat \[page 275\]](#) or [TomEE 7 \[page 272\]](#)) context path to the URL, it's automatically appended.

## Example

If you configure `/test_context_path` as a context path, and the Web application is available on `/test_app`, then:

When you call:	You'll be redirected to:
<code>&lt;HOST&gt;:&lt;PORT&gt;/test_app</code>	<code>&lt;HOST&gt;:&lt;PORT&gt;/test_context_path/test_app</code>

The default context path value for Tomcat and TomEE 7 is " " (empty string).

For more information on how to change the default value, see:

- [Tomcat 9 \[page 275\]](#)
- [Tomcat 10 \[page 278\]](#)
- [TomEE 7 \[page 272\]](#)

### 4.1.3.5.1.7 Multitenant Java Applications

In the Cloud Foundry environment, you can develop and run multitenant Java applications, and share them with multiple consumers simultaneously on SAP BTP.

#### What is Multitenancy?

SAP BTP provides a multitenant functionality that allows application providers to own, deploy, and operate tenant-aware applications for multiple consumers, with reduced costs. For example, the application provider can upgrade the application for all your consumers instead of performing each update individually, or can share resources across multiple consumers. The application consumers launch the applications using consumer-specific URLs, and can configure certain application features.

To learn more, see: [Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#)

#### Multitenancy in the SAP Java Buildpack

SAP Java Buildpack provides the possibility for multitenant applications running on Cloud Foundry to consume tenant-aware data sources out of the box. This is achieved by integrating the SAP Service Manager capabilities in the buildpack. The SAP Java buildpack provides out of the box tenant-aware data sources for:

- All tenants that have already been onboarded to an SAP Service Manager service instance.
- Newly onboarded tenants at runtime. No restart of the Java application is needed.

## Configure the Application Multitenancy

To achieve multitenant support in your application deployed with the SAP Java buildpack, your application should meet the following requirements:

1. Multitenancy support for Cloud Foundry is set up in advance. See: [Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#)
2. The application should be bound to a managed database service. Your application should also use one of the following datasources that the SAP Java buildpack provides:
  - com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory
  - com.sap.xs.jdbc.datasource.tomee7.TomEE7DataSourceFactorySee section: [Configure Tenant-Aware Data Source \[page 309\]](#)
3. The security concept that the application uses should be XSUAA. See section: [Configure XSUAA Authentication Method \[page 310\]](#)

Once these requirements are fulfilled, the application takes care of the onboarding of tenants in the SAP Service Manager service they use. For each request that comes from the application, the SAP Java buildpack will acquire (obtain) the tenant ID from the request through the JWT token provided by the XSUAA service. The buildpack will provide different database instance object for each tenant.

## Configure Tenant-Aware Data Source

Once a multitenant application is bound to a managed database service instance, provisioning of the tenant-aware data source comes out of the box.

In the following example of **context.xml**, the **jpa-db-managed** service instance is used to configure the data source. Provided that the **jpa-db-managed** service instance is of type **managed-hana** or **service-manager**, it will make the **jdbc/DatasourceOne** tenant aware.

```
<?xml version='1.0' encoding='utf-8'?>
<Context>
  <Resource name="jdbc/DatasourceOne"
    auth="Container"
    type="javax.sql.DataSource"
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"
    service="jpa-db-managed"
    personalize="true"/>
</Context>
```

If a custom tenant provider is used, define the class in the Resource configuration as a **tenantProvider**. For example:

```
<?xml version='1.0' encoding='utf-8'?>
<Context>
  <Resource name="jdbc/DatasourceOne"
    auth="Container"
    type="javax.sql.DataSource"
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"
    service="jpa-db-managed"
    personalize="true"/>
    tenantProvider="com.sap.test.custom.provider.UserDefinedTenantProvider"/>
</Context>
```

**Tip:** If you want a **non-JPA** application to consume `jdbc/DatasourceOne`, use the following code:

```
public class TestServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
    @Resource(name = "jdbc/DatasourceOne")  
    private DataSource tenantAwareDataSource;  
    ...  
}
```

## Configure XSUAA Authentication Method

The configuration is done in the `login-config` section of the `web.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>  
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance xmlns="http://java.sun.com/xml/ns/javaee xsi:schemaLocation="http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd version="3.0">  
    <display-name>sample</display-name>  
    <login-config>  
        <auth-method>XSUAA</auth-method>  
    </login-config>  
</web-app>
```

## Control Determination of the Current Tenant for Tenant-Aware Data Source

As described above, by default the tenant-aware data source determines which tenant to be used for a specific request by getting the current log user from the JWT token provided by the XSUAA service.

For specific scenarios, the application can get control over the tenant determination with the following steps:

1. Implement and bundle a class which will handle the tenant provisioning. For example:

```
package com.sap.test.custom.provider;  
public class UserDefinedTenantProvider implements Supplier<String> {  
    private static ThreadLocal<String> currentTenant = new  
InheritableThreadLocal<>();  
    public static ThreadLocal<String> getCurrentTenant() {  
        currentTenant.set("tenant1");  
        return currentTenant;  
    }  
    @Override  
    public String get() {  
        return getCurrentTenant().get();  
    }  
}
```

### ① Note

The class is **required** in order to implement the `Supplier` interface.

2. Define the application class as a `tenantProvider`. For example:

```
<?xml version='1.0' encoding='utf-8'?>  
<Context>  
    <Resource name="jdbc/DatasourceOne"
```

```

auth="Container"
type="javax.sql.DataSource"
factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"
service="jpa-db-managed"
personalize="true"/>
tenantProvider="com.sap.test.custom.provider.UserDefinedTenantProvider"/>
</Context>
```

## Related Information

[Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 377\]](#)

### 4.1.3.5.1.8 Multiple Buildpack Framework

The Multiple Buildpack Framework enables SAP Java Buildpack to act as the final buildpack in a multiple buildpack deployment. It reads the contributions of other, earlier buildpacks and incorporates them into its standard staging.

#### ⓘ Note

If you try to use SAP Java Buildpack as a non-final buildpack, an error will be thrown.

When SAP Java Buildpack acts as the final buildpack in a multiple buildpack deployment, it honors the following core contract integration points with non-final buildpacks:

Integration Point	Buildpack Usage
/bin	An existing /bin directory contributed by a non-final buildpack will be added to the <code>&lt;\$PATH&gt;</code> of the application as it executes.
/lib	An existing /lib directory contributed by a non-final buildpack will be added to the <code>&lt;\$LD_LIBRARY_PATH&gt;</code> of the application as it executes.

### 4.1.3.5.1.9 Logging and Tracing

#### Context

SAP Java Buildpack integrates the [Cloud Foundry Java Logging Support](#) libraries, allowing applications to produce logs in the JSON format that can be parsed by the SAP Application Logging service for SAP BTP.

## Procedure

- Bind the application to the SAP Application Logging service. This way, you can produce application logs and forward them to a central application logging stack.
- If you want to generate request metrics for applications using a Tomcat or a TomEE 7 application container, add `com.sap.hcp.cf.logging.servlet.filter.RequestLoggingFilter` to the `web.xml` file.

```
<filter-name>request-logging</filter-name>
  <filter-class>com.sap.hcp.cf.logging.servlet.filter.RequestLoggingFilter</
filter-class>
</filter>
<filter-mapping>
  <filter-name>request-logging</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

### ⓘ Note

The default log level of `com.sap.hcp.cf.logging.servlet.filter.RequestLoggingFilter` is set to INFO.

- Check the logs of the application.

If the binding of your application to the SAP Application Logging service is successful, the application logs can be found on `https://logs.cf.<LANDSCAPENAME>`. To request the logs of your application, run:

```
cf logs <app_name> --recent
```

## Related Information

[Configure a Java Application for Logs and Traces \[page 312\]](#)

[SAP Application Logging Service for Cloud Foundry](#)

[Spring Boot: Logging ↗](#)

## 4.1.3.5.1.9.1 Configure a Java Application for Logs and Traces

Configure the collection of log and trace messages generated by a Java application in SAP BTP, Cloud Foundry.

### Prerequisites

- You have bound an application to the SAP Application Logging service. See: [Logging and Tracing \[page 311\]](#)
- You have installed and configured Maven.
- For **SAP Java Buildpack 1**: You use SAP JVM 8 or SapMachine 11 or 17, and your `<JAVA_HOME>` environment variable points to this location.

- For **SAP Java Buildpack 2**: You use SapMachine 17 or 21, and your <JAVA\_HOME> environment variable points to this location.
- You do **not** have any SLF4J and logback JAR files in the application.

### Caution

The JARs for the [SLF4J](#) and [logback](#) are included as part of the Tomcat and TomEE 7 runtimes provided by SAP Java Buildpack. Packing them in the application can cause problems during class loading.

## Context

The recommended framework for logging is Simple Logging Facade for Java (SLF4J). To use this framework, you can create an instance of the [org.slf4j.Logger](#) and [org.slf4j.LoggerFactory](#) classes. You can use SLF4J to configure your Java application to generate logs and traces, and if appropriate – set the logging and tracing levels.

### Note

It is the application's responsibility to ensure that [logback](#) is configured in a secure way in the case when the application overrides the default logback configuration included in SAP Java Buildpack. See step 2 below.

## Procedure

1. Instruct Maven that the application should not package the SLF4J dependency, since it's already provided by the runtime.

To do that, go to the `pom.xml` file and set scope **provided** for the following dependency:

- For SAP Java Buildpack 1:

```
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.36</version>
  <scope>provided</scope>
</dependency>
```

- For SAP Java Buildpack 2:

```
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>2.0.13</version>
  <scope>provided</scope>
</dependency>
```

2. (Optional) Configure the logback in case the application overrides the default logback configuration. To do that, you need to do the following steps:

- Under path `META-INF/sap_java_buildpack/resources/tomcat/conf`, create a custom `logback.xml` file. For example:

#### ❖ Example

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration debug="false" scan="false">
    <contextListener
        class="ch.qos.logback.classic.jul.LevelChangePropagator"/>
    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder class="${encoder}">
            ${pattern}
        </encoder>
    </appender>
    <root level="INFO">
        <appender-ref ref="STDOUT" />
    </root>
</configuration>
```

- Create a `resource_configuration.yml` file or add the following entries to your existing `resource_configuration.yml` file (if using Tomcat):

#### ❖ Example

```
---
tomcat/conf/logback.xml:
  pattern: ''
  encoder: ''
```

**NOTE:** You can also add some default values for `pattern` and `encoder`.

- Add the following property in your `manifest.yml` file. For example (if using Tomcat):

#### ❖ Example

```
---
applications:
- name: <APP_NAME>
  ...
  env:
    JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/logback.xml' :
      {'encoder': 'com.sap.hcp.cf.logback.encoder.JsonEncoder', 'pattern':
      '' } ]"
```

## Related Information

[Configure the Access Logs of an Application \[page 315\]](#)

## 4.1.3.5.1.9.2 Configure the Access Logs of an Application

SAP Java Buildpack uses the *logback-access* module to provide HTTP-access log functionality.

### Prerequisites

- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
- [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)

### Context

The access logs differ slightly from the other logs and traces. There are many standard tools available that read access logs from any server, display them, calculate statistics and so on. That's why the format of the access log produced by Tomcat and TomEE 7 is not SAP-specific. The default pattern defined in the configuration file is "%date %r %s %b".

To find the access log files of your application, you have to connect via SSH to the SAP BTP, Cloud Foundry container and go to the `logs` directory.

To change the default access log settings, follow the steps below.

### Procedure

1. Modify the default configuration.

SAP Java Buildpack uses LogbackValve to configure access logs in Tomcat and TomEE 7. The following environment variables are used in the configuration file and can be overwritten:

- `<ACCESS_LOG_FILE>`
- `<ACCESS_LOG_FILE_COUNT>`
- `<ACCESS_LOG_FILE_MAX_SIZE>`
- `<logback.access.log.xs.pattern>`

2. (Optional) Change the name of the access log files.

By default, access logs are placed in the application container in the `logs/access.log` file. You can configure the directory containing this file, as well as the name of the files by using environment variable `ACCESS_LOG_FILE`. To do that, define the full path to the desired directory and the file name. For example:

```
env:  
  ACCESS_LOG_FILE: "/home/vcap/logs/access-log/access.log"
```

3. (Optional) Change the number of access log files.

You can change the number of stored files with the environment variable `ACCESS_LOG_FILE_COUNT`. The default value is **3**. For example:

```
env:
```

```
ACCESS_LOG_FILE_COUNT: 5
```

4. (Optional) Change the size of the access log files.

You can modify the size of every access log file with the environment variable `ACCESS_LOG_FILE_MAX_SIZE`. The default value is **8** MB. For example:

```
env:  
  ACCESS_LOG_FILE_MAX_SIZE: 10
```

5. (Optional) Change the format of the HTTP access log

You can change the default format and have your access logs written differently. SAP Java Buildpack uses the [resource configuration](#) functionality to allow this. Changing the `pattern` attribute in the configuration file of Tomcat and TomEE 7 is supported. The application has to provide the new value via the `JBP_CONFIG_RESOURCE_CONFIGURATION` mechanism.

- For SAP Java Buildpack 1:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/logback-access-  
  localhost.xml':{ 'logback.access.log.xs.pattern' : 'combined' } ]"
```

- For SAP Java Buildpack 2:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/conf/logback-access-  
  localhost.xml':{ 'logback.access.log.pattern' : 'combined' } ]"
```

**NOTE:** If you use an Apache TomEE container, the configuration is the same – just replace `tomcat` with `tomee`.

## 4.1.3.5.1.10 Debugging Java Applications

Debugging an application helps you detect and diagnose errors in your code.

You can control the execution of your program by setting breakpoints, suspending threads, stepping through the code, and examining the contents of the variables. See:

- [Debug an Application Running on SAP JVM \[page 317\]](#)
- [Debug an Application Running on SapMachine \[page 318\]](#)
- [Debug a Community Java Buildpack Application \[page 319\]](#)

## 4.1.3.5.1.10.1 Debug an Application Running on SAP JVM

You can debug an application running on a Cloud Foundry container that uses SAP JVM.

### Prerequisites

- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
- [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)
- Deploy your application using SAP Java Buildpack. To do that, from the cf CLI execute:

```
cf push <app name> -p <war file> -b sap_java_buildpack
```
- Ensure that SSH is enabled for the application. See [Accessing Apps with SSH](#) ↗

#### ⓘ Note

SAP JVM is included in SAP Java Buildpack 1. With SAP JVM, you can enable debugging on-demand without the need to set any debugging parameters, hence without having to restart the application or SAP JVM.

### Context

After enabling the debugging port, you need to open an SSH tunnel that connects to this port.

### Procedure

1. To enable debugging or to check the debugging state of your JVM, run `jvmmon` in your Cloud Foundry container by executing:

```
cf ssh <app name> -c "app/META-INF/.sap_java_buildpack/sapjvm/bin/jvmmon"
```

2. From the `jvmmon` command line window, execute:

```
start debugging
```

3. (Optional) To confirm that debugging is enabled and see which port is open, execute:

```
print debugging information
```

The following is an example of the information displayed by `jvmmon`:

```
State: Debugging back is waiting for debugger to connect
Port: 8000
Client:
Globally accessible
```

The default port is 8000.

4. To exit jvmmon, execute: `q`
5. To open the SSH tunnel, execute:

```
cf ssh <app name> -N -T -L 8000:127.0.0.1:8000
```

Your local port 8000 is connected to the debugging port 8000 of the JVM running in the Cloud Foundry container. Also, 8000 is the default port.

#### ⓘ Note

The connection is active until you close the SSH tunnel. When you finish debugging, close the SSH tunnel by pressing `[Ctrl]` + `[C]`.

6. Connect a Java debugger to your application. For example, use the standard Java debugger provided by Eclipse IDE and connect to `localhost:8000`

## 4.1.3.5.1.10.2 Debug an Application Running on SapMachine

Debug a Java web application running on a Cloud Foundry container that uses SapMachine.

### Prerequisites

- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
- [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)

### Context

To debug an application using [SapMachine](#), you need to open a debugging port on your Cloud Foundry container, and then open an SSH tunnel to connect to that port.

### Procedure

1. To open the debugging port, you need to configure the `JAVA_OPTS` parameter in your JVM. By default, the `agentlib` (which enables the debugging) is pre-configured. It could be turned off by setting the following property in the environment:

```
env
...
JBP_CONFIG_SAP_MACHINE_JDK: "[default_debug_agent_active: false]"
```

You could also disable the default configuration by specifying it manually. Set the following option in the application manifest file:

```
JBP_CONFIG_JAVA_OPTS: "[ java_opts: '-agentlib:jdwp=transport=dt_socket,address=8000,server=y,suspend=n' ]"
```

You can change the default port 8000 in the command.

2. To enable SSH tunneling in your application, execute:

```
cf enable-ssh
```

3. To open the SSH tunnel, execute:

```
cf ssh <app name> -N -T -L 8000:127.0.0.1:8000
```

Your local port 8000 is connected to the debugging port 8000 of the JVM, which is running in the Cloud Foundry container.

#### ⓘ Note

The connection is active until you close the SSH tunnel. When you finish debugging, close the SSH tunnel by pressing **Ctrl** + **C**.

4. Connect a Java debugger to your application. For example, use the standard Java debugger provided by Eclipse IDE and connect to `localhost:8000`.

### 4.1.3.5.1.10.3 Debug a Community Java Buildpack Application

You can debug an application running on a Cloud Foundry container that uses the community Java buildpack.

#### Prerequisites

- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
- [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)

#### Context

To debug an application using the [community Java buildpack](#), you need to open a debugging port on your Cloud Foundry container and open an SSH tunnel that will connect to that port.

## Procedure

1. To open the debugging port, you need to configure the JAVA\_OPTS parameter in your JVM. From the cf CLI, execute:

```
cf set-env <app name> JAVA_OPTS '-Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8000'
```

2. To enable SSH tunneling for your application, execute:

```
cf enable-ssh <app name>
```

3. To activate the previous changes, restart the application by executing:

```
cf restart <app name>
```

4. To open the SSH tunnel, execute:

```
cf ssh <app name> -N -T -L 8000:127.0.0.1:8000
```

Your local port 8000 is connected to the debugging port 8000 of the JVM running in the Cloud Foundry container. Also, 8000 is the default port.

### ⓘ Note

The connection is active until you close the SSH tunnel. When you finish debugging, close the SSH tunnel by pressing **Ctrl** + **C**.

5. Connect a Java debugger to your application. For example, use the standard Java debugger provided by Eclipse IDE and connect to `localhost:8000`.

## 4.1.3.5.1.11 Profiling Java Applications

Application profiling helps you analyze the resource consumption of your Java applications running on virtual machine - SAP JVM or SapMachine.

### Related Information

[Profiling an Application Running on SAP JVM \[page 321\]](#)

[Profiling an Application Running on SapMachine \[page 323\]](#)

## 4.1.3.5.1.11.1 Profiling an Application Running on SAP JVM

The SAP JVM Profiler is a tool that helps you analyze the resource consumption of a Java application running on SAP Java Virtual Machine (JVM). You can use it to profile simple standalone Java programs or complex enterprise applications.

### Prerequisites

- Install the SAP JVM Tools for Eclipse. For more information, see [Install the SAP JVM Tools in Eclipse \[page 322\]](#).
- Open a debugging connection using an SSH tunnel. For more information, see [Debug an Application Running on SAP JVM \[page 317\]](#)

### Context

To measure resource consumption, the SAP JVM Profiler enables you to run different profiling analyses. Each profiling analysis creates profiling events that focus on different aspects of resource consumption. For more information about the available analyses, see the SAP JVM Profiler documentation in Eclipse. Go to ► [Help](#) ▶ [Help Contents](#) ▶ [SAP JVM Profiler](#) ▶.

### Procedure

1. In Eclipse, open the *Profiling* perspective.
2. From the *VM Explorer* view, choose  (*Connect Host Via Debugging Port*).
3. Enter your host name and port number, and choose *Next*.

#### Note

Your port number is your local SSH tunnel endpoint.

4. Choose the analysis you want to run and specify your profiling parameters.

#### Note

To use the thread annotation filters, complete the fields under the *Analysis Options* section. By default, all filters are set to \*, which means that all annotations pass the filter.

## 4.1.3.5.1.11.1.1 Install the SAP JVM Tools in Eclipse

*SAP JVM Profiler* is included in the SAP JVM Tools.

### Prerequisites

You have downloaded and installed **Eclipse IDE for Java Developers**. See: [Eclipse Packages](#) ↗

### Procedure

1. Open the Eclipse IDE and navigate to .
2. In the *Work with* combo box, enter <https://tools.hana.ondemand.com/oxygen> and choose *Add*.
3. Select and choose *Next*.
4. Select *SAP JVM Tools*, accept the terms and conditions, and choose *Finish*.
5. Restart the Eclipse IDE to apply the new changes.

### Results

After the restart, a new *SAP JVM Profiler* section is displayed in the *Overview* page.

## 4.1.3.5.1.11.1.2 (Optional) Add your SAP JVM to the VM Explorer

You can add your SAP JVM to the *VM Explorer* view of the **SAP JVM Profiler**.

### Prerequisites

- Install the SAP JVM Tools in Eclipse [page 322].
- Download and Install the Cloud Foundry Command Line Interface [page 2452]
- Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface [page 2453]

## Context

From the [VM Explorer](#), you can debug, monitor or profile your SAP JVM. For more information about the [VM Explorer](#), see the [SAP JVM Profiler](#) documentation in the Eclipse Help.

### ⓘ Note

You need to use the **jvmmond** tool, which is included in the SAP Java Buildpack.

## Procedure

1. From the cf CLI, execute:

```
cf ssh app name -c "app/META-INF/.sap_java_buildpack/sapjvm/bin/jvmmond -port 50003 -J-Dcom.sap.jvm.tools.portRange=50004-50005 -J-Djava.rmi.server.hostname=127.0.0.1"
```

This starts the **jvmmond** service on your Cloud Foundry container. It is listening to port [50003](#). This command also specifies a port range of 50004-50005 in case additional ports need to be opened.

2. To enable an SSH tunnel for these ports, execute:

```
cf ssh <app name> -N -T -L 50003:127.0.0.1:50003 -L 50004:127.0.0.1:50004 -L 50005:127.0.0.1:50005
```

3. In your Eclipse IDE, open the [Profiling](#) perspective, and from the [VM Explorer](#) view choose [Manage Hosts](#).
4. Choose [Add](#) and enter the following host name and port number: **localhost:50003**

## Results

Your application is added to the VM Explorer.

## 4.1.3.5.1.11.2 Profiling an Application Running on SapMachine

You can use Java Flight Recorder ([JFR](#)) to profile your Java application on SapMachine, and Java Mission Control ([JMC](#)) to do remote profiling and analysis.

## Prerequisites

- You have downloaded and installed a JDK Mission Control console. To do that, go to [SapMachine](#) → [Download](#) → [JMC 9](#), select your operating system, and then choose [Download](#).

- You have installed the Cloud Foundry command line interface. See: [Download and Install the cf CLI](#)
- You are logged on to a SAP BTP, Cloud Foundry space. See: [Log On to the Cloud Foundry Environment Using cf CLI](#)
- You have a Java application that is up and running on SAP BTP, Cloud Foundry.
- You have enabled SSH for your application. See: [Configuring SSH access at the app level](#) ↗

## Context

To profile with JMC, you need to start the Management Agent on a RMI port and then open an SSH tunnel to connect to that port. To do that, follow the steps below.

### ⓘ Note

In the steps below, we use **myapp** and **5555** as *exemplary* application name and port. Replace them with your actual app name and a free port number on your localhost.

## Procedure

1. Set the RMI server to `localhost`. To do that, set the following `JBP_CONFIG_JAVA_OPTS` option in the `manifest.yml` file of your application:

```
---
applications:
- name: myapp
  buildpack: sap_java_buildpack
...
env:
  TARGET_RUNTIME: tomcat
  JBP_CONFIG_COMPONENTS: "jres:
    ['com.sap.xs.java.buildpack.jre.SAPMachineJRE']"
  JBP_CONFIG_SAP_MACHINE_JRE: '{version: 11.+}'
  JBP_CONFIG_JAVA_OPTS: "[java_opts: '-
Djava.rmi.server.hostname=127.0.0.1']"
```

2. Start the Management Agent by running `jcmd` in your Cloud Foundry container. Execute:

```
cf ssh myapp -c "app/META-INF/.sap_java_buildpack/sap_machine_jre/bin/
jcmd $(pgrep java) ManagementAgent.start jmxremote.authenticate=false
jmxremote.ssl=false jmxremote.port=5555 jmxremote.rmi.port=5555"
```

Depending on what your SapMachine is using (JRE or JDK), specify the path accordingly (`sap_machine_jre` or `sap_machine_jdk`). To learn more, see: [SapMachine \[page 290\]](#)

### Result:

```
7:
Command executed successfully
```

Management Agent is started and is listening to port **5555**.

3. (Optional) You can check the status of the Management Agent. Execute:

```
cf ssh myapp -c "app/META-INF/.sap_java_buildpack/sap_machine_jre/bin/jcmd $(pgrep java) ManagementAgent.status"
```

4. Enable an SSH tunnel for this port. Execute:

```
cf ssh myapp -N -T -L 5555:127.0.0.1:5555
```

Your *local* port 5555 is connected to the *remote* port 5555 of the JVM, which is running in the Cloud Foundry container.

#### Note

The connection is active until you close the SSH tunnel. When you finish profiling, close the SSH tunnel by pressing **[Ctrl] + [C]**.

5. Connect with JMC to `localhost:5555`. To do that:
  1. Open your JDK Mission Control (JMC) console.
  2. From the *JVM Browser* tab, choose icon *Create a new custom JVM connection*.
  3. For *Port*, enter **5555** (or your actual port number), and then choose *Finish*.
6. In the left-side menu, a new node *localhost:5555* appears.
7. Double-click on *MBean Server*.
8. To start profiling, go to *Flight Recorder* and from its context menu, choose *Start Flight Recording*.
9. Set up the event settings. From the drop-down menu, choose one of the following options (templates):
  - *Continuous* – Low overhead configuration, safe for continuous use in production environments
  - *Profiling* – Low overhead configuration for profiling
  - *gc* – GC related events. Generates a small recording size.
  - *gc\_details* – GC related events. Gets heap statistic and generates a large recording size.
  - Create your own configuration. To do that, choose *Template Manager*, then select a predefined template and choose *Duplicate*. Rename the new template and edit its configuration parameters according to your needs and preferences.
10. After profiling is done, go to *localhost:5555* and from its context menu, choose *Disconnect*.
11. Then go back to your command line and close the SSH tunnel by pressing **[Ctrl] + [C]**.
12. Stop the Management Agent. Execute:

```
cf ssh myapp -c "app/META-INF/.sap_java_buildpack/sap_machine_jre/bin/jcmd $(pgrep java) ManagementAgent.stop"
```

## Related Information

[Debug an Application Running on SapMachine \[page 318\]](#)

## 4.1.3.5.1.12 Configuring a Database Connection

You can configure your application to use a database connection so that the application can persist its data. This configuration is applicable for the [Tomcat 9 \[page 275\]](#), [TomEE 7 \[page 272\]](#), and [Tomcat 10 \[page 278\]](#) application containers.

- [Configure a Database Connection for the Tomcat Application Container \[page 328\]](#)
- [Configure a Database Connection for the TomEE 7 Application Container \[page 326\]](#)
- [Database Connection Configuration Details \[page 329\]](#)
- [SAP HANA HDI Data Source Usage \[page 331\]](#)

### 4.1.3.5.1.12.1 Configure a Database Connection for the TomEE 7 Application Container

#### Procedure

1. Create a `resources.xml` file.

##### ⓘ Note

If the data source is to be used from a Web application, you have to create the file inside the `WEB-INF` / directory.

If the data source is to be used from Enterprise JavaBeans (EJBs), you have to create the file inside the `META-INF` / directory.

The `resources.xml` file has to be inside the application's WAR file and has to contain information about the data source to be used.

```
<?xml version='1.0' encoding='utf-8'?>
<resources>
  <Resource id="jdbc/DefaultDB" provider="xs.openejb:XS Default JDBC Database"
    type="javax.sql.DataSource" >
    service=${service_name_for_DefaultDB}
  </Resource>
</resources>
```

2. Add the default keys and their values.

You need to include the data source information in `META-INF/sap_java_buildpack/config/resource_configuration.yml` of the WAR file.

##### ⚡ Sample Code

Defining a default service in `resource_configuration.yml` for a Web application

```
---
tomee7/webapps/ROOT/WEB-INF/resources.xml:
```

```
service_name_for_DefaultDB: di-core-hdi
```

### ↔ Sample Code

Defining a default service in resource\_configuration.yml for an EJB

```
---
```

```
tomee7/webapps/ROOT/META-INF/resources.xml:
```

```
  service_name_for_DefaultDB: di-core-hdi
```

3. Add the key values to be updated.

Include the data source information to be updated in the manifest.yml file.

### ↔ Sample Code

Defining a new service for the look-up of the data source in a Web application

```
env:
```

```
  TARGET_RUNTIME: tomee7
```

```
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee7/webapps/ROOT/WEB-INF/
```

```
  resources.xml': { 'service_name_for_DefaultDB' : 'my-local-special-di-core-
```

```
  hdi' } ]"
```

### ↔ Sample Code

Defining a new service for the look-up of the data source in an EJB

```
env:
```

```
  TARGET_RUNTIME: tomee7
```

```
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee7/webapps/ROOT/META-INF/
```

```
  resources.xml': { 'service_name_for_DefaultDB' : 'my-local-special-di-core-
```

```
  hdi' } ]"
```

## Results

As a result of this configuration, when the application starts, the `com.sap.xs.jdbc.datasource.TomEEDataSourceFactory` factory takes the parameters bound to the my-local-special-di-core-hdi service from the environment, creates a data source, and binds it under `jdbc/DefaultDB`. The application then uses the Java Naming and Directory Interface (JNDI) to look up how to connect with the database.

## Related Information

[SAP HANA HDI Data Source Usage \[page 331\]](#)

[Database Connection Configuration Details \[page 329\]](#)

[Configuring a Database Connection \[page 326\]](#)

## 4.1.3.5.1.12.2 Configure a Database Connection for the Tomcat Application Container

### Procedure

1. Create a context.xml file.

The context.xml file has to be inside the META-INF/ folder of the application WAR file and has to contain information about the data source to be used.

```
<?xml version='1.0' encoding='utf-8'?>
<Context>
  <Resource name="jdbc/DefaultDB"
    auth="Container"
    type="javax.sql.DataSource"
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"
    service="${service_name_for_DefaultDB}" />
</Context>
```

2. Add the default keys and their values.

You need to include the data source information in META-INF/sap\_java\_buildpack/config/resource\_configuration.yml of the WAR file.

```
---
tomcat/webapps/ROOT/META-INF/context.xml:
  service_name_for_DefaultDB: di-core-hdi
```

3. Add the key values to be updated.

You include the data source information to be updated in manifest.yml.

```
env:
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/webapps/ROOT/META-INF/
context.xml' : { 'service_name_for_DefaultDB' : 'my-local-special-di-core-
hdi' } ]"
```

### Results

When the application starts, the factory named com.sap.xs.jdbc.datasource.TomcatDataSourceFactory takes the parameters bound for service *my-local-special-di-core-hdi* from the environment, creates a data source, and binds it under jdbc/DefaultDB. The application then uses the Java Naming and Directory Interface (JNDI) to look up how to connect with the database.

## 4.1.3.5.1.12.3 Database Connection Configuration Details

Define details of the database connection used by your Java Web Application running on SAP BTP, Cloud Foundry with SAP Java Buildpack.

### Configuration Files

To configure your application to establish a connection to the SAP HANA database, you must specify details of the connection in a configuration file. For example, you must define the data source that the application will use to discover and look up data. The application then uses a Java Naming and Directory Interface (JNDI) to look up the specified data in the file.

The easiest way to define the required data source is to declare the keys for the data source in a resource file.

For the Tomcat application container, you can create a `context.xml` file in the `META-INF` / directory with the following content:

#### ↔ Sample Code

`context.xml`

```
<?xml version='1.0' encoding='utf-8'?>
<Context>
  <Resource name="jdbc/DefaultDB"
    auth="Container"
    type="javax.sql.DataSource"
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"
    service="di-core-hdi"/>
</Context>
```

For the TomEE 7 application container, you can create a `resources.xml` file in the `WEB-INF` / directory with the following content:

#### ↔ Sample Code

`resources.xml`

```
<?xml version='1.0' encoding='utf-8'?>
<resources>
  <Resource id="jdbc/DefaultDB" provider="xs.openejb:XS Default JDBC Database"
    type="javax.sql.DataSource" >
    service=di-core-hdi
  </Resource>
</resources>
```

The problem with this simple approach is that your WAR file cannot be signed, and any modifications can only be made in the WAR file. For this reason, it is recommended that you do not use the method in a production environment but rather take advantage of the [Resource Configuration \[page 305\]](#) feature of the SAP Java Buildpack. Use modification settings in `resource_configuration.yml` and `manifest.yml` as illustrated in the following examples.

Defining a default service in `resource_configuration.yml`.

## ↔ Sample Code

resource\_configuration.yml

```
---  
tomcat/webapps/ROOT/META-INF/context.xml:  
  service_name_for_DefaultDB: di-core-hdi
```

Specifying a default name for a service is useful (for example, for automation purposes) only if you are sure there can be no conflict with other names. For this reason, it is recommended that you include a helpful and descriptive error message instead of a default value. That way the error message will be part of an exception triggered when the data source is initialized, which helps troubleshooting.

Sample resource\_configuration.yml.

## ↔ Sample Code

```
---  
tomcat/webapps/ROOT/META-INF/context.xml:  
  service_name_for_DefaultDB: Specify the service name for Default DB in  
manifest.yml via "JBP_CONFIG_RESOURCE_CONFIGURATION" ..
```

## Placeholders

The generic mechanism JBP\_CONFIG\_RESOURCE\_CONFIGURATION basically replaces the key values in the specified files. For this reason, if you use placeholders in the configuration files, it is important to ensure that you use unique names for the placeholders, see [Resource Configuration \[page 305\]](#).

Sample context.xml.

## ↔ Sample Code

context.xml

```
<?xml version='1.0' encoding='utf-8'?>  
<Context>  
  <Resource name="jdbc/DefaultDB"  
    auth="Container"  
    type="javax.sql.DataSource"  
    description="Datasource for general functionality"  
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"  
    service="${service_name_for_DefaultDB}"  
    maxActive="${max_Active_Connections_For_DefaultDB}" />  
  <Resource name="jdbc/DefaultXADB"  
    auth="Container"  
    type="javax.sql.XADataSource"  
    description="Datasource for functionality requiring more than one  
    transactional resource"  
    factory="com.sap.xs.jdbc.datasource.tomcat.TomcatDataSourceFactory"  
    service="${service_name_for_DefaultXADB}"  
    maxActive="${max_Active_Connections_For_DefaultXADB}" />  
</Context>
```

The names of the defined placeholders are also used in the other resource files.

Sample `resource_configuration.yml`.

#### ↔ Sample Code

`resource_configuration.yml`

```
---
tomcat/webapps/ROOT/META-INF/context.xml:
  service_name_for_DefaultDB: di-core-hdi
  max_Active_Connections_For_DefaultDB: 100
  service_name_for_DefaultDB: di-core-hdi-xa
  max_Active_Connections_For_DefaultXADB: 100

env:
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/webapps/ROOT/META-INF/
context.xml' : { 'service_name_for_DefaultDB' : 'my-local-special-di-
core-hdi' , 'max_Active_Connections_For_DefaultDB' : '30' ,
'service_name_for_DefaultXADB' : 'my-local-special-xa-di-core-hdi' ,
'max_Active_Connections_For_DefaultXADB' : '20' } ]"
```

Sample `manifest.yml`.

#### ↔ Sample Code

`manifest.yml`

```
env:
  JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomcat/webapps/ROOT/META-INF/
context.xml' : { 'service_name_for_DefaultDB' : 'my-local-special-di-
core-hdi' , 'max_Active_Connections_For_DefaultDB' : '30' ,
'service_name_for_DefaultXADB' : 'my-local-special-xa-di-core-hdi' ,
'max_Active_Connections_For_DefaultXADB' : '20' } ]"
```

## 4.1.3.5.1.12.4 SAP HANA HDI Data Source Usage

If you want your Java application to consume SAP HANA Database (for example, to use an HDI container), follow the steps below.

### Procedure

1. Create a service instance for the HDI container, using the `cf create-service` command.

```
cf create-service hana hdi-shared my-hdi-container
```

2. Bind the service instance to a Java application.

Specify the service instance in the Java application's deployment [`manifest.yml`](#) file.

#### ↔ Sample Code

```
services:
- my-hdi-container
```

3. Add a resource reference XML block. Its technical name must be the same as the name of the service instance.

#### ↔ Sample Code

```
<resource-ref>
  <res-ref-name>jdbc/my-hdi-container</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
</resource-ref>
```

Add this resource reference to the following files, respectively:

- For Tomcat: `web.xml` and `context.xml`
  - For TomEE 7: `web.xml` and `resource.xml`
4. Look up the data source in your code.

You can find the reference to the data source in two ways – by using annotations or with JNDI look-up. Add the relevant code to your Java servlet:

- Annotations

```
@Resource(name = "jdbc/my-hdi-container")
private DataSource ds;
```

- Java Naming and Directory Interface (JNDI) look-up

```
Context ctx = new InitialContext();
return (DataSource) ctx.lookup("java:comp/env/jdbc/my-hdi-container");
```

## Related Information

[SAP HDI Containers](#)

[Set Up a Schema or an HDI Container \(Cloud Foundry\)](#)

## 4.1.3.5.1.13 Dynatrace Integration

Dynatrace OneAgent is a Java agent that sends all captured monitoring data to your Dynatrace monitoring environment for analysis.

## Context

Dynatrace supports full-stack monitoring for Cloud Foundry – from the application to the infrastructure layer. To integrate it with your Java application, follow the steps below.

## Procedure

1. Obtain an [environment ID](#) and an [API access token](#).
2. Create a user-provided service.

The service name should contain the string "dynatrace" (for example, **dynatraceservice**). The `environmentid` and `apitoken` parameters have been generated in the previous step. You also need to set up the `apiurl`, which depends on the Dynatrace type you want to integrate.

- **SaaS Dynatrace**

```
"apiurl": "https://<environmentid>.live.dynatrace.com/api"
```

- **Managed Dynatrace**

```
"apiurl": "https://<your_domain>/e/<environmentid>/api"
```

Then, in cf CLI, run the following command:

```
cf cups dynatraceservice -p "environmentid, apitoken, apiurl"
```

For more information, see:

- Dynatrace Docs: [Create a user-provided service in your SAP BTP, Cloud Foundry Environment](#)
- Cloud Foundry Docs: [User-Provided Service Instances](#)

3. Bind the application to the service instance created in the previous step, by using the `manifest.yml` file.

### Sample Code

```
---
applications:
- name: myapp
  memory: 1G
  instances: 1
  buildpack: sap_java_buildpack
  services:
    - dynatraceservice
```

**NOTE:** If there are more than one Dynatrace services bound to the application, the service that's going to be used is the one that was chronologically bound first.

4. Deploy the application to the SAP BTP, Cloud Foundry environment with the updated `manifest.yml` file. Run:

```
cf push myapp
```

Once the application receives web traffic, monitoring data will show up in Dynatrace.

## Related Information

[Dynatrace Docs: Access tokens API](#)

[Dynatrace Docs: Cloud Foundry Monitoring](#)

## 4.1.3.5.1.14 SAP Connectivity ApiExt

Use the SAP Connectivity ApiExt feature (provided by SAP Java Buildpack) to retrieve destination configurations or authentication headers ready to be used against the remote target system.

### ⚠ Restriction

SAP Connectivity ApiExt is only available for Tomcat application containers included in SAP Java Buildpack.

Also, Spring Boot works with SAP Connectivity ApiExt only when using WAR deployment.

### ⓘ Note

SAP Connectivity ApiExt is **not** the recommended way for consuming the Destination service as it only covers a subset of the service's APIs. It's primarily meant to be used internally by [SAP Java Connector](#) (SAP JCo) and for Neo migration scenarios, where there are limited possibilities to modify the code.

The **recommended** consumption mechanism for Java applications is [SAP Cloud SDK for Java](#) or using the [Destination service REST API](#).

## Prerequisites

To use SAP Connectivity ApiExt during runtime, you first need to create:

- A service instance for the Destination service
- A destination entity in the Destination service that you want to retrieve from the application. See: [Managing Destinations](#)
- (Optional) A service instance for the Connectivity service – only needed if you plan to use it for destinations of type `ProxyType=OnPremise`

## Activate the API

1. Open the `pom.xml` file of your Tomcat-based Java application and add the following dependency:

```
<dependency>
    <groupId>com.sap.cloud.connectivity.apiext</groupId>
    <artifactId>com.sap.cloud.connectivity.apiext</artifactId>
    <version>${connectivity-apiext.version}</version>
    <scope>provided</scope>
</dependency>
```

2. Now open the `manifest.yml` file and set environment variable `USE_CONNECTIVITY_APIEXT` to `true`:

```
---
applications:
- name: <APP_NAME>
  ...
env:
  TARGET_RUNTIME: tomcat
```

```
USE_CONNECTIVITY_APIEXT: true
```

3. Bind the previously created service instances to your application. To do that, add their names at the end of the `manifest.yml` file. For example:

```
---
applications:
- name: <APP_NAME>
  ...
env:
  TARGET_RUNTIME: tomcat
  USE_CONNECTIVITY_APIEXT: true
services:
  - my_destination
  - my_connectivity
  ...
```

For more information, see [ConnectivityConfiguration API: Procedure](#)

## Related Information

[Create Destinations from Scratch](#)

[SAP Java Connector \[page 335\]](#)

### 4.1.3.5.1.15 SAP Java Connector

You can use the SAP Java Connector through the SAP Java buildpacks.

## Activation

SAP Java Buildpack (1 and 2) provide an option to use [SAP Java Connector \(SAP JCo\)](#).

To activate SAP JCo in the buildpack, set the environment variable `USE_JCO` to `true`.

#### ↔ Sample Code

`manifest.yml`

```
---
applications:
- name: <app_name>
  ...
env:
  USE_JCO: true
```

The previous method of activating the SAP JCo feature, by defining a service instance for both the connectivity and destination services in the `manifest.yml` file, has been deprecated and stops working at the end of a transition phase.

The activation of SAP JCo provides all relevant libraries in the application container, so that [SAP JCo API](#) can be used. To use SAP JCo during runtime, a destination and an SAP Authorization and Trust Management (XSUAA) service instance are mandatory. Additionally, if you want to set up on-premise connectivity, a connectivity service instance is required.

## Limitations

SAP JCo is only available for the Tomcat-based application containers that are included in the SAP Java buildpacks. Spring Boot applications only work with SAP JCo if you're using WAR deployment.

## Related Information

[Invoking ABAP Function Modules via RFC](#)

[Invoke ABAP Function Modules in On-Premise ABAP Systems](#)

## 4.1.3.5.1.16 SAP Java Buildpack Libraries

On this page you can find changelogs about the latest updates made in the main five libraries of SAP Java Buildpack 1 and 2, provided on Maven Repository.

### XS Env

Use this library to set up and access environment variables and services.

**Technical name:** `xs-env`

#### ⓘ Note

- In SAP Java Buildpack 1, `xs-env` is compiled to Java 8 bytecode.
- In SAP Java Buildpack 2, `xs-env` is compiled to Java 17 bytecode.

**Repository:** <https://mvnrepository.com/artifact/com.sap.cloud.sjb/xs-env>

## Changelog:

SAP Java Buildpack 1	SAP Java Buildpack 2
( 29 May 2024 )	( 29 May 2024 )
<b>Version 1.53.0</b>	<b>Version 2.6.0</b>
<ul style="list-style-type: none"><li>A new <i>findServices</i> functionality is introduced. It provides a list with Cloud Foundry services that match certain required conditions.</li></ul>	<ul style="list-style-type: none"><li>A new <i>findServices</i> functionality is introduced. It provides a list with Cloud Foundry services that match certain required conditions.</li></ul>
( 14 May 2024 )	( 14 May 2024 )
<b>Version 1.52.0</b>	<b>Version 2.5.0</b>
<ul style="list-style-type: none"><li><i>com.fasterxml.jackson</i> is updated from version 2.17.0 to 2.17.1</li><li><i>com.fasterxml.jackson.databind</i> is updated from version 2.17.0 to 2.17.1</li><li>The URL and description of the <i>xs-env</i> library have been improved.</li></ul>	<ul style="list-style-type: none"><li><i>com.fasterxml.jackson</i> is updated from version 2.17.0 to 2.17.1</li><li><i>com.fasterxml.jackson.databind</i> is updated from version 2.17.0 to 2.17.1</li><li>The URL and description of the <i>xs-env</i> library have been improved.</li></ul>
( 02 April 2024 )	( 20 March 2024 )
<b>Version 1.51.0</b>	<b>Version 2.4.0</b>
<ul style="list-style-type: none"><li><i>xs-java-parent</i> is updated from version 1.16.0 to 1.17.0</li></ul>	<ul style="list-style-type: none"><li><i>com.fasterxml.jackson</i> is updated from version 2.16.1 to 2.17.0</li><li><i>com.fasterxml.jackson.databind</i> is updated from version 2.16.1 to 2.17.0</li></ul>
( 20 March 2024 )	( 16 February 2024 )
<b>Version 1.50.0</b>	<b>Version 2.2.0</b>
<ul style="list-style-type: none"><li><i>com.fasterxml.jackson</i> is updated from version 2.16.1 to 2.17.0</li><li><i>com.fasterxml.jackson.databind</i> is updated from version 2.16.1 to 2.17.0</li></ul>	<ul style="list-style-type: none"><li>This is the first publicly released version.</li><li><i>com.fasterxml.jackson</i> is updated from version 2.16.0 to 2.16.1</li><li><i>com.fasterxml.jackson.databind</i> is updated from version 2.16.0 to 2.16.1</li></ul>

## XS User Holder

Use this library to access user information during authentication and authorization when using SAP BTP Security Services Integration Libraries.

**Technical name:** `xs-user-holder`

### ⓘ Note

- In SAP Java Buildpack 1, `xs-user-holder` is compiled to Java 8 bytecode.

- In SAP Java Buildpack 2, `xs-user-holder` is compiled to Java 17 bytecode.

**Repository:** <https://mvnrepository.com/artifact/com.sap.cloud.sjb/xs-user-holder> ↗

**Changelog:**

SAP Java Buildpack 1	SAP Java Buildpack 2
(14 May 2024)	(14 May 2024)
<b>Version 1.56.0</b>	<b>Version 2.5.0</b>
<ul style="list-style-type: none"> <li>• <code>sap.cloud.security.version</code> is updated from 2.17.3 to 2.17.5</li> <li>• The URL and description of the <code>xs-user-holder</code> library have been improved.</li> </ul>	<ul style="list-style-type: none"> <li>• The URL and description of the <code>xs-user-holder</code> library have been improved.</li> </ul>
(02 April 2024)	(02 April 2024)
<b>Version 1.55.0</b>	<b>Version 2.4.0</b>
<ul style="list-style-type: none"> <li>• <code>com.sap.cloud.sjb.xs-env.version</code> is updated from 1.50.0 to 1.51.0</li> </ul>	<ul style="list-style-type: none"> <li>• For <code>sap.cloud.security.version</code> 3.3.5, an obsolete <code>java-api</code> dependency has been removed.</li> </ul>
(23 January 2024)	(05 March 2024)
<b>Version 1.54.0</b>	<b>Version 2.3.0</b>
<ul style="list-style-type: none"> <li>• <code>sap.cloud.security.version</code> is updated from 2.17.2 to 2.17.3</li> </ul>	<ul style="list-style-type: none"> <li>• <code>sap.cloud.security.version</code> is updated from 3.3.1 to 3.3.5</li> </ul>

## XS HDB Conn Options

Use this library to handle SAP HANA DB connection options.

**Technical name:** `xs-hdb-conn-options`

### ⓘ Note

- In SAP Java Buildpack 1, `xs-hdb-conn-options` is compiled to Java 8 bytecode.
- In SAP Java Buildpack 2, `xs-hdb-conn-options` is compiled to Java 17 bytecode.

**Repository:** <https://mvnrepository.com/artifact/com.sap.cloud.sjb/xs-hdb-conn-options> ↗

## Changelog:

SAP Java Buildpack 1	SAP Java Buildpack 2
( 29 May 2024 )	( 29 May 2024 )
<b>Version 1.53.0</b>	<b>Version 2.7.0</b>
<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 1.52.0 to 1.53.0</li></ul>	<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 2.5.0 to 2.6.0</li></ul>
( 14 May 2024 )	( 14 May 2024 )
<b>Version 1.52.0</b>	<b>Version 2.6.0</b>
<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 1.51.0 to 1.52.0</li><li>• The URL and description of the <i>xs-hdb-conn-options</i> library have been improved.</li></ul>	<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 2.4.0 to 2.5.0</li><li>• The URL and description of the <i>xs-hdb-conn-options</i> library have been improved.</li></ul>
( 02 April 2024 )	( 02 April 2024 )
<b>Version 1.51.0</b>	<b>Version 2.5.0</b>
<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 1.50.0 to 1.51.0</li></ul>	<ul style="list-style-type: none"><li>• Minor changes</li></ul>
( 20 March 2024 )	( 20 March 2024 )
<b>Version 1.50.0</b>	<b>Version 2.4.0</b>
<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 1.49.0 to 1.50.0</li></ul>	<ul style="list-style-type: none"><li>• <i>com.sap.cloud.sjb.xs-env.version</i> is updated from 2.3.0 to 2.4.0</li></ul>

## XS Java Parent

Use this library to consume common dependencies and Maven plug-in configurations.

**Technical name:** xs-java-parent

### ⓘ Note

- In SAP Java Buildpack 1, `xs-java-parent` is compiled to Java 8 bytecode.
- In SAP Java Buildpack 2, `xs-java-parent` is compiled to Java 17 bytecode.

**Repository:** <https://mvnrepository.com/artifact/com.sap.cloud.sjb/xs-java-parent> ↗

## Changelog:

SAP Java Buildpack 1	SAP Java Buildpack 2
(14 May 2024)	(14 May 2024)
<b>Version 1.18.0</b>	<b>Version 2.3.0</b>
<ul style="list-style-type: none"><li>The URL and description of the <a href="#">xs-java-parent</a> library have been improved.</li></ul>	<ul style="list-style-type: none"><li>The URL and description of the <a href="#">xs-java-parent</a> library have been improved.</li></ul>
(02 April 2024)	(16 February 2024)
<b>Version 1.17.0</b>	<b>Version 2.2.0</b>
<ul style="list-style-type: none"><li>Minor changes</li></ul>	<ul style="list-style-type: none"><li>This is the first publicly released version.</li><li><a href="#">xs-memory-calculator</a> has been removed from the <code>pom.xml</code> file.</li><li><a href="#">xs-sapjvm-statistics-provider</a> has been removed from the <code>pom.xml</code> file.</li></ul>

## XS JDBC Routing Datasource

This wrapper library handles datasource creation in single or multitenancy mode while working with JDBC.

**Technical name:** `xs-jdbc-routing-datasource`

### ⓘ Note

- In SAP Java Buildpack 1, `xs-jdbc-routing-datasource` is compiled to Java 8 bytecode.
- In SAP Java Buildpack 2, `xs-jdbc-routing-datasource` is compiled to Java 17 bytecode.

**Repository:** <https://mvnrepository.com/artifact/com.sap.cloud.sjb/xs-jdbc-routing-datasource>

## Changelog:

SAP Java Buildpack 1	SAP Java Buildpack 2
(14 May 2024)	(14 May 2024)
<b>Version 1.47.0</b>	<b>Version 2.5.0</b>
<ul style="list-style-type: none"><li>The URL and description of the <a href="#">xs-jdbc-routing-datasource</a> library have been improved.</li><li>The eager loading of data sources has been deleted for all managed instances.</li></ul>	<ul style="list-style-type: none"><li>The URL and description of the <a href="#">xs-jdbc-routing-datasource</a> library have been improved.</li></ul>

SAP Java Buildpack 1	SAP Java Buildpack 2
( 02 April 2024 )	( 02 April 2024 )
<b>Version 1.46.0</b>	<b>Version 2.4.0</b>
<ul style="list-style-type: none"> <li>• Minor changes</li> </ul>	<ul style="list-style-type: none"> <li>• Minor changes</li> </ul>
( 20 February 2024 )	( 20 March 2024 )
<b>Version 1.45.0</b>	<b>Version 2.3.0</b>
<ul style="list-style-type: none"> <li>• <code>com.sap.cloud.instancemanager.version</code> is updated from 2.2.1 to 3.14.0. Also, its <code>client-api</code> dependency has been changed to just <code>client</code>.</li> </ul>	<ul style="list-style-type: none"> <li>• Minor changes</li> </ul>

## Related Information

Maven Repository: SAP Cloud SJB 

### 4.1.3.5.1.17 Application-Provided Library Components

SAP Java Buildpack can automatically provide JDBC drivers for SAP HANA and PostgreSQL in the classpath if the respective service instance is bound.

The application can also provide an own version of these drivers in the `WEB-INF/lib` directory. In this case, SAP Java Buildpack will not download the current version of the driver but place the application-provided version in the classpath. The drivers have to follow the respective file name pattern:

- `ngdbc*.jar`
- `postgresql*.jar`

#### Note

When using this mechanism, the application developer is responsible to keep the drivers up-to-date.

## Auxiliary Library Components

The application developer can provide additional components alongside the aforementioned libraries. For example, this can be a custom SSL factory for the PostgreSQL JDBC Driver, which has to be accessible to the class loader that loads the actual library.

To do so, the additional libraries have to be placed in a JAR file in the `WEB-INF/lib` directory. The JAR file has to follow the respective file name pattern:

- `sjb-ngdbc-aux*.jar`
- `sjb-postgresql-aux*.jar`

Any number of files following these patterns can be placed in the `WEB-INF/lib` directory and will be handled by SAP Java Buildpack.

This mechanism works with both application-provided and buildpack-provided libraries.

## 4.1.3.5.1.18 Bill of Materials (BOM)

For Maven projects, the versions of the SAP Java Buildpack dependencies and the APIs provided by supported runtime containers can be consumed through a Bill of Materials (BOM).

The BOM contains the dependencies you can rely on in a runtime environment provided by SAP Java Buildpack. It can also be used to control the versions of a project's dependencies without the need to manually maintain the version of each single artifact.

The BOMs of SAP Java Buildpack are provided through [Maven Central](#).

## Usage

There are two runtimes for which a BOM is provided: Tomcat and TomEE 7. The version of the BOM must match the version of the `sap_java_buildpack` you're using. For example, if your application uses `sap_java_buildpack` version 1.75.0, the BOM must be in version 1.75.0 as well.

If you want to use version control, add the desired BOM artifact to your project's `pom.xml` file in the section `dependencyManagement`. The following code is an example.

- For Tomcat:

```
...
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.sap.cloud.sjb.cf</groupId>
      <artifactId>cf-tomcat-bom</artifactId>
      <version>1.75.0</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
...
```

- For TomEE 7:

```
...
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.sap.cloud.sjb.cf</groupId>
```

```

<artifactId>cf-tomee7-bom</artifactId>
<version>1.75.0</version>
<type>pom</type>
<scope>import</scope>
</dependency>
</dependencies>
</dependencyManagement>
...

```

To add the dependencies that you want, use *groupId* and *artifactId*:

```

...
<dependency>
    <groupId>com.sap.cloud.sjb</groupId>
    <artifactId>xs-env</artifactId>
</dependency>
<dependency>
    <groupId>com.sap.cloud.sjb</groupId>
    <artifactId>xs-user-holder</artifactId>
</dependency>
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-annotations</artifactId>
</dependency>
<dependency>
    <groupId>org.slf4j</groupId>
    <artifactId>jul-to-slf4j</artifactId>
</dependency>
<dependency>
    <groupId>ch.qos.logback</groupId>
    <artifactId>logback-access</artifactId>
</dependency>
...

```

If you simply want to have all dependencies in your project, you can also add the BOM to your project's section. By doing so, you don't need to enumerate single dependency artifacts.

## Related Information

[Maven Repository: SAP Java Buildpack BOM](#)

### 4.1.3.5.2 Developing Node.js in the Cloud Foundry Environment

This section offers selected information for Node.js development on SAP BTP, Cloud Foundry and references to more detailed sources.

You'll get information about the buildpack supported by SAP, the Node.js packages, and how to consume them in your application.

There is also a tutorial with an introduction to securing your application, and some tips and tricks for developing and running Node.js applications on SAP BTP, Cloud Foundry.

## Node.js Community Buildpack

SAP BTP uses the standard [Node.js buildpack](#) provided by the Cloud Foundry community to deploy Node.js applications.

To get familiar with the buildpack and how to deploy applications with it, take a look at the [Cloud Foundry Node.js Buildpack documentation](#).

## SAP Node.js Packages

You can download and consume SAP developed Node.js packages via the SAP NPM Registry. There is an overview of Node.js packages developed by SAP, what they are meant for, and where they are included in the SAP HANA Developer Guide for XS Advanced Model. See:

- [Download and Consume SAP Node.js Packages](#)
- [Standard Node.js Packages](#)
- [The SAP NPM Registry](#)

## Buildpack Versioning

The SAP BTP, Cloud Foundry environment provides one recent version of `nodejs_buildpack` as part of its system buildpacks. To check this version, proceed as follows:

1. Log in to a particular SAP BTP region and subaccount. For example, if your region is **eu10**, run:

```
cf api https://api.cf.eu10.hana.ondemand.com
```

2. Then run:

```
cf buildpacks
```

To learn about changes in the Node.js buildpack's versions and features, regularly check the latest [buildpack releases](#) in the GitHub community page.

## Supported Versions

The `nodejs_buildpack` running on SAP BTP, Cloud Foundry environment supports the following versions:

- Node.js **18**
- Node.js **20**
- Node.js **22**

### ⚠ Restriction

Bear in mind that the following listed SAP libraries don't support Node.js 22 yet. Thus, if an application running on Node.js 22 depends on any of them, it will fail during deployment or redeployment:

- sap/xsenv
- sap/textbundle
- sap/xss-secure
- sap/e2e-trace
- sap/audit-logging
- sap/instance-manager

## Deprecated Versions

Node.js 16 reached end of life on **September 11, 2023** and was removed from the Cloud Foundry community in [version 1.8.15](#). This means that deployment and redeployment of applications running on Node.js 16 will **fail**. For more information, see: [Node.js Roadmap](#)

We recommend that you migrate your applications to Node.js **18** or **20** as soon as possible.

### ⓘ Note

Applications using XSJS are strongly impacted as the [@sap/fibers](#) library (on which [@sap/xsjs](#) depends) is **not supported** on Node.js 16 and later versions. To learn more, see: [Migrating Applications from XSJS to Async-XSJS \[page 346\]](#)

In exceptional cases (if you haven't completed the migration to Node.js 18), to avoid application failures during redeployment, you may pin the last buildpack version that contains Node.js 16, as provided by the [nodejs-buildpack](#) community. To learn how, see: [Specify a buildpack version in manifest.yml \[page 353\]](#)

### → Remember

SAP does **not** recommend use of deprecated Node.js versions, as support and security fixes are no longer provided for them.

## What's New

To check the latest news and updates about the Node.js buildpack, go to its release notes: [What's New for Node.js Buildpack](#)

### ⓘ Note

In May 2023, SAP migrated the root file system used in the Cloud Foundry environment in SAP BTP from the deprecated `cflinuxfs3` stack to `cflinuxfs4`. If you are running Node.js applications on SAP BTP, Cloud Foundry using the Node.js buildpack, we recommend that you update and migrate your applications, as

well as the Node.js buildpack. For more information about migration timelines, risks, and consequences, see:

- [Deprecation of Cloud Foundry Stack cflinuxfs3 and Migration to cflinuxfs4](#)
- [What's New: cflinuxfs4 becomes the default stack](#)

## Troubleshooting

If you encounter an issue while using the Node.js buildpack, you can:

- Search for your problem in our Guided Answers: [Node.js Buildpack](#)
- Create an incident for your specific problem, using support component **BC-CP-CF-BLDP**. To provide the necessary details, use the following template: [Initial Problem-Related Data](#)

## Node.js Tutorial

The following tutorial will guide you through creating a Node.js application in Cloud Foundry Command Line Interface (cf CLI), consuming a Cloud Foundry service, and setting up authentication and authorization checks. See: [Create an Application with Cloud Foundry Node.js Buildpack](#)

## Tips and Tricks

For selected tips and tricks for your Node.js development, see [Tips and Tricks for Node.js Applications \[page 351\]](#).

### 4.1.3.5.2.1 Migrating Applications from XSJS to Async-XSJS

This page explains how to migrate XSJS to Async-XSJS to benefit from latest Node.js versions (16 and later).

## Context

SAP HANA XSJS applications are based on *synchronous* API. On the other hand, Node.js is an *asynchronous* runtime. Thus, for XSJS applications that run on Node.js there is a certain code incompatibility. Up until Node.js 14, this incompatibility has been handled by an NPM package called [@sap/fibers](#) (forked from [laverdet/node-fibers](#)). Unfortunately, [@sap/fibers](#) is not compatible with Node.js version 16 and higher.

Also, Node.js supports two kinds of scripts:

- CommonJS - it's synchronous and does not support top-level await. In XSJS, all files are loaded as common Java scripts.
- ECMAScript - it's fully asynchronous and supports top-level await

The only possible way for your XSJS applications to be up and running on latest Node.js versions is to modify their codes – by migrating them to a new XSJS layer with asynchronous API. See the migration process below to learn what you need to do.

### ⓘ Note

In exceptional cases (if you haven't completed the migration to Async-XSJS), to avoid application failures during redeployment, you may pin the last buildpack version that contains Node.js14, as provided by the [nodejs-buildpack](#) community. You can do this in your **manifest.yml** file, and then redeploy your app. To learn how, see: [Specify a buildpack version in manifest.yml \[page 353\]](#)

Please be advised, that SAP does **not** recommend usage of Node.js 14, as no support and security fixes are provided for this version anymore.

## Migration Process

1. To switch from XSJS to Async-XSJS, open your `package.json` file, and in the **dependencies** section, add the [@sap/async-xsjs](#) package. For example:

```
{  
  "name": "myapp",  
  "engines": {  
    "node": "16.x.x"  
  },  
  ...  
  "dependencies": {  
    "express": "^4.18.1",  
    "@sap/xssec": "^3.2.15",  
    "@sap/async-xsjs": "^1.0.0"  
  }  
  ...  
}
```

2. Enable the ECMAScript virtual-machine interface on application startup.

With [@sap/async-xsjs](#), your asynchronous JavaScript code is loaded by default as ECMAScript (ES). To make use of the support that Node.js provides for ES modules, you need to start your application using the option `--experimental-vm-modules`.

```
{  
  "name": "myapp",  
  "engines": {  
    "node": "16.x.x"  
  },  
  ...  
  "dependencies": {  
    "express": "^4.18.1",  
    "@sap/xssec": "^3.2.15",  
    "@sap/async-xsjs": "^1.0.0"  
  }  
  "scripts": {  
    "start": "node --experimental-vm-modules index.js"  
  }  
}
```

```
    }, ...  
    ...  
}
```

3. In all your `.xsjs` and `.xsjslib` files, for every asynchronous function call, add an **await** statement.
4. Every function that uses an **await** statement must be declared as **async**.
5. XSJS files should be loaded as ECMAScript. To do this, in the end of every function in an `.xsjslib` file, add an **export** statement.

#### ⓘ Note

You can do this for `.xsjs` files too, but that would be only necessary if a function needs to be exported for some reason (like job execution or job handling).

6. If you're using the [@sap/audit-logging](#) package in your XSJS application, the code must be migrated. For example, if you have a code snippet like this:

```
auditLog  
  .securityMessage('Content of the message')  
  .by($.session.getUsername())  
  .sync.log();
```

it should be converted to:

```
await auditLog  
  .securityMessage('Content of the message')  
  .by($.session.getUsername())  
  .log();
```

To do these steps, refer to the API of the new [@sap/async-xsjs](#) package. See: [SAP HANA 2 - ASYNC XS JavaScript API Reference](#)

You can find this API reference if you go to the [SAP HANA Platform](#) product page, and then choose [View All](#) → [SAP HANA Asynchronous XS JavaScript API Reference](#).

#### ⓘ Note

For huge application codes, you can optimize your work by using a migration tool. See: [@sap/async-migrator](#)

Bear in mind that this is an open-source product and a fully-covered migration of your XSJS code is not guaranteed.

## Examples

The following exemplary code snippet from an `.xsjslib` file:

#### ⚡ Example

Original Code

```
...  
$.import('lib', 'converters');  
function select(tableName){
```

```

let conn;
try {
  conn = $.hdb.getConnection();
  let resultSet = conn.executeQuery('SELECT * FROM "' + tableName + '"');
  return $.lib.converters.resultSet_to_Entities(resultSet, ["id", "name"]);
} finally {
  if (conn) {
    conn.commit();
    conn.close();
  }
}
}

```

should be converted to:

### ❖ Example

Modified Code

```

...
await $.import('lib', 'converters');
async function select(tableName){
  let conn;
  try {
    conn = await $.hdb.getConnection();
    let resultSet = await conn.executeQuery('SELECT * FROM "' + tableName +
  ''');
    return $.lib.converters.resultSet_to_Entities(resultSet, ["id", "name"]);
  } finally {
    if (conn) {
      await conn.commit();
      conn.close();
    }
  }
}
export default {select};

```

## Side Effects

### .sync Statements

Since the `@sap/fibrous` package is not supported anymore, the `.sync` statements will no longer work. To fix a code that uses the `sync` property, see **step 4** from the procedure above.

### this Pointer

Since all `.xsjs` and `.xsjslib` files are based on ECMAScript module, `this` pointer behaves differently from when used with CommonJS. For example, when ECMAScript is loaded, the value of `this` is `undefined`.

In the context of a module function execution, the value of pointer `this` is the `export <object>` module.

The following sample code will throw an error due to `this=undefined` during loading.

### ❖ Example

```

==== my.xsjslib ===XSJS ===

this.funcA = function(){}

```

```
(function(self){
    self.funcB = function(){} ;
})(this);
this.x = 5;
```

Adapt the code to Async-XSJS:

#### • Example

```
==== my.xsjslib ====Async-XSJS ====

function funcA(){}
function funcB(){}
var x = 5;
export default {funcA, funcB, x};
```

This example just illustrates the problem and gives a possible solution. Depending on the software design of your application, there could be various solutions.

To learn more about the differences between these two kinds of Node.js scripts, see:

- [ECMAScript documentation](#)
- [Differences between ES modules and CommonJS](#)

**Array.forEach** and **Array.map**

In the Async-XSJS layer, the **Array.forEach** and **Array.map** functions change their behavior when the argument function is asynchronous.

For example, the SQL statements are read and executed sequentially, one after the other, and then written to the database in the same order.

#### • Example

```
==== my.xsjslib ====XSJS ===

var sqls = ['create table "PRODUCT"("ID" integer, "NAME": varchar(200))',
            'insert into "PRODUCT" values (1, \'iphone\')'
          ];
function runSql(sql){
  var conn = $.hdb.getConnection();
  conn.execute(sql);
}
sqls.forEach(runSql);
...
<additional code>
```

The following example shows how the same code looks when adapted to Async-XSJS:

#### • Example

```
==== my.xsjslib ====Async-XSJS ===

var sqls = ['create table "PRODUCT"("ID" integer, "NAME": varchar(200))',
            'insert into "PRODUCT" values (1, \'iphone\')'
          ];
async function runSql(sql){
  var conn = await $.hdb.getConnection();
  await conn.execute(sql);
}
```

```
sqls.forEach(runSql);
...
<some additional code>
```

In the previous example, the SQL statements are only scheduled for execution; they are not executed immediately. When it is time to record them in the database, the order of operation might be different to the order in which the tables were initially read.

#### ⚠ Caution

If you have additional code after the `sqls.forEach` function, it will be executed before any of the SQL statements.

To avoid unwanted side effects and ensure that the SQL statements are created in the database in the correct order, you can replace `sqls.forEach(runSql);` with one of the following examples:

- ⚡ Example

##### *Sequential execution*

```
for (int i=0; i<sqls.length; i++){
    await runSql(sqls[i]);
}
<some additional code>
//SQL statements are executed in the correct order, then the additional
code is executed
```

- ⚡ Example

##### *Parallel execution*

```
await Promise.all(sqls.map(runSql));
<some additional code>
//SQL statements are executed in a random order, then the additional code
is executed
```

The same configuration steps can be applied for `Array.map` in analogical way – just replace `sqls.forEach` with `sqls.map`.

Another function that can be asynchronous is `Array.reduce`.

## 4.1.3.5.2.2 Tips and Tricks for Node.js Applications

Get to know the Node.js buildpack for the SAP BTP, Cloud Foundry environment.

Check the following Cloud Foundry documentation:

- [Node.js Buildpack](#)
- [Tips for Node.js Developers](#)

## NPM Version

When deploying an application in the SAP BTP, Cloud Foundry environment without specifying the *npm* version in the **package.json** file, it will install the default *npm* version (depending on the Node.js version) during the build step. A version mismatch can cause the build step to fail.

To check your Node.js version, run:

```
node --version
```

To check your npm version, run:

```
npm --version
```

### → Tip

We recommend that you use the same *npm* version as the one on the SAP BTP, Cloud Foundry environment.

Alternatively, you can specify the *npm* version in the **package.json** file. For example, if you use Node.js 20, you can set:

```
"engines": {  
  "npm": "^8.4.1"  
},
```

### ⓘ Note

Bear in mind that **Node.js** and **npm** are different products, with their own independent versions. To check the version mapping between them, see [NodeJS: Previous Releases](#).

## Vendor Application Dependencies

Sometimes, productive Cloud Foundry applications might need to be deployed as self-contained. That means, they need to carry all of their dependencies so that the staging process does not require any network calls. For more information, see: [Vendor App Dependencies](#).

Depending on the region where the application is deployed:

- Running on the China (Shanghai) region: it is **mandatory** for the application to be deployed as self-contained.
- Running on the AWS, Azure, or GCP regions: it is only **recommended** but not mandatory for the application to be deployed as self-contained.

There are various reasons to use vendoring. For example, productive applications usually rely on security scans and because *npm* doesn't provide reliable dependency fetch, it's possible that your Node.js application ends up with different dependencies in case such are installed during deployment. Additionally, *npm* downloads any missing dependencies from its registry, so if this registry isn't accessible for some reason, the deployment can fail.

## ⓘ Note

Bear in mind when using dependencies containing native code that you need to reinstall in the same environment as the Cloud Foundry container, or make sure that the package has built-in support for it.

To ensure that prepackaged dependencies are pushed to the SAP BTP, Cloud Foundry environment and on-premise runtime, make sure that the `node_modules` directory isn't listed in the `.cfignore` file. It's also preferable that development dependencies are not deployed for productive deployments. To ensure that, run:

```
npm prune --production
```

## Out of Memory at Runtime

For performance reasons, Node.js (powered by V8 engine) has lazy garbage collection. Even if there are no memory leaks in your application, this can cause occasional restarts, as explained in the official Cloud Foundry documentation – [Tips for Node.js Applications](#).

Enforce the garbage collector to run before the memory is consumed by limiting the V8 application's heap size at about ~75% of the available memory. To do that, you can either use the `OPTIMIZE_MEMORY` environment variable supported by the Node.js buildpack, or specify the V8 heap size directly in your application `start` command (recommended).

Example for an application started with 256M of RAM:

### ↗ Sample Code

```
node --max_old_space_size=192 server.js
```

You can optimize V8 behavior using additional options. To list these options, run:

```
node --v8-options
```

## Specify a buildpack version in `manifest.yml`

At some point, you might need (or decide) to deploy your application with a particular buildpack version from the community [nodejs-buildpack](#) repository. For example, if this buildpack contains a Node.js version that is no longer supported by the SAP BTP, Cloud Foundry environment.

Let's say, you want to pin version [1.8.9](#). To do that, proceed as follows:

1. Open the `manifest.yml` file of your Node.js application.
2. For the `buildpack` attribute, add the URL to version 1.8.9, like this:

```
---  
applications:  
- name: myapp  
  random-route: true  
  buildpack: https://github.com/cloudfoundry/nodejs-buildpack.git#v1.8.9
```

```
memory: 128M
```

3. Redeploy your application. Run:

```
cf push myapp
```

#### Alternative way:

If you don't want to make changes in your **manifest.yml** file, you can include the buildpack version in the `cf push` command.

- To deploy just a **single** application with this particular buildpack version, run:

```
cf push myapp -b https://github.com/cloudfoundry/nodejs-buildpack.git#v1.8.9
```

- To pin this buildpack version for **all** applications running in your SAP BTP, Cloud Foundry environment subaccount, run:

```
cf push -b https://github.com/cloudfoundry/nodejs-buildpack.git#v1.8.9
```

### Specify application memory in `manifest.yml`

When deploying an application in the Cloud Foundry environment without specifying the application memory requirements, the Cloud Foundry controller assigns the default (1G of RAM currently) for your application. Many Node.js applications require less memory and assigning the default is a waste of resources.

To save memory from your quota, specify the memory size in the deployment descriptor of your Cloud Foundry application – `manifest.yml`. For example:

```
---  
applications:  
- name: myapp  
  memory: 128M  
  buildpack: nodejs_buildpack  
...
```

### 4.1.3.5.3 Developing Python in the Cloud Foundry Environment

This section offers selected information for Python development on the SAP BTP, Cloud Foundry environment and references to more detailed sources.

You'll get information about the buildpack supported by SAP, about the Python packages, and how to consume them in your application.

There is also a tutorial with an introduction to securing your application, and some tips and tricks for developing and running Python applications on the SAP BTP, Cloud Foundry environment.

## Python Community Buildpack

SAP BTP uses the standard [Python buildpack](#) provided by the Cloud Foundry community to deploy Python applications.

To get familiar with the buildpack and how to deploy applications with it, take a look at the [Cloud Foundry Python Buildpack documentation](#).

## SAP Python Packages

SAP includes a selection of Python packages, which are available for download and use for customers and partners who have the appropriate access authorization. To download them, log on to [SAP Software Download Center](#) and search for software component **XS PYTHON**, which is an archive that contains the SAP packages.

The following table lists the SAP Python packages that are currently available. For more details about the contents of each Python package, as well as any configuration tips, see the README file in the corresponding package.

Package	Description
sap_instance_manager	Python package for creating and deleting service instances per tenant within an application at runtime.
sap_audit_logging	Provides audit logging functionalities for Python applications.
sap_xssec	SAP XS Advanced Container Security API for Python.
sap_cf_logging	This is a collection of support libraries for Python applications running on Cloud Foundry that: <ul style="list-style-type: none"><li>• provide means to emit structured application log messages</li><li>• instrument web applications of your application stack to collect request metrics</li></ul>
hdbcli	The SAP HANA Database Client provides means for database connectivity.

## Buildpack Versioning

The SAP BTP, Cloud Foundry environment provides one recent version of `python_buildpack` as part of its system buildpacks. To check this version, proceed as follows:

1. Log in to a particular SAP BTP region and subaccount. For example, if your region is **eu10**, run:

```
cf api https://api.cf.eu10.hana.ondemand.com
```

2. Then run:

```
cf buildpacks
```

To learn about changes in the Python buildpack's versions and features, regularly check the latest [buildpack releases](#) in the GitHub community page.

## Supported Versions

The `python_buildpack` supports the following versions:

- Python **3.8**
- Python **3.9**
- Python **3.10**
- Python **3.11**

### ⓘ Note

Version **3.7** is out of maintenance, as per [Python release roadmap](#). We strongly recommend that you switch to version 3.8 or higher.

You can also decide to deploy your application with a particular buildpack version from the community [python-buildpack](#) repository. To learn how, see: [Specify a buildpack version in manifest.yml \[page 358\]](#)

### → Remember

SAP does **not** recommend use of deprecated Python versions, as support and security fixes are no longer provided for them.

## What's New

To check the latest news and updates about the Python buildpack, go to its [buildpack releases](#) on the GitHub community page.

### ⓘ Note

In May 2023, SAP migrated the root file system used in the Cloud Foundry environment in SAP BTP from the deprecated `cflinuxfs3` stack to `cflinuxfs4`. If you are running Python applications on SAP BTP, Cloud

Foundry using the Python buildpack, we recommend that you update and migrate your applications, as well as the Python buildpack. For more information about migration timelines, risks, and consequences, see:

- [Deprecation of Cloud Foundry Stack cflinuxfs3 and Migration to cflinuxfs4](#)
- [What's New for SAP BTP, Cloud Foundry Runtime \(cflinuxfs\)](#)

## Troubleshooting

If you encounter an issue while using the Python buildpack, you can:

- Search for your problem in our Guided Answers: [Python Buildpack](#)
- Create an incident for your specific problem, using support component **BC-CP-CF-BLDP**. To provide the necessary details, use the following template: [Initial Problem-Related Data](#)

## Python Tutorial

The following tutorial will guide you through creating a Python application in Cloud Foundry Command Line Interface (cf CLI), consuming Cloud Foundry services, and setting up authentication and authorization checks. See: [Create an Application with Cloud Foundry Python Buildpack](#)

## Tips and Tricks

Selected tips and tricks for your Python development. See [Tips and Tricks for Python Applications \[page 357\]](#).

### 4.1.3.5.3.1 Tips and Tricks for Python Applications

Get to know the Python buildpack for the SAP BTP, Cloud Foundry environment.

Check the following Cloud Foundry documentation: [Python Buildpack](#)

## Vendor Application Dependencies

Sometimes, productive Cloud Foundry applications might need to be deployed as self-contained. That means, they need to carry all of their dependencies so that the staging process does not require any network calls. For more information, see: [Vendor App Dependencies](#)

Depending on the region where the application is deployed:

- Running on the China (Shanghai) region: it is **mandatory** for the application to be deployed as self-contained

- Running on the AWS, Azure, or GCP regions: it is only **recommended** but not mandatory for the application to be deployed as self-contained.

## Specify a buildpack version in `manifest.yml`

At some point, you might need (or decide) to deploy your application with a particular buildpack version from the community [python-buildpack](#) repository. For example, if this buildpack contains a Python version that is no longer supported by the SAP BTP, Cloud Foundry environment.

Let's say, you want to pin version [1.8.4](#). To do that, proceed as follows:

1. Open the **manifest.yml** file of your Python application.
2. For the `buildpack` attribute, add the URL to version 1.8.4, like this:

```
---
applications:
- name: myapp
  random-route: true
  buildpack: https://github.com/cloudfoundry/python-buildpack.git#v1.8.4
  memory: 128M
```

3. Redeploy your application. Run:

```
cf push myapp
```

### Alternative way:

If you don't want to make changes in your **manifest.yml** file, you can include the buildpack version in the `cf push` command.

- To deploy just a single application with this particular buildpack version, run:

```
cf push myapp -b https://github.com/cloudfoundry/python-buildpack.git#v1.8.4
```

- To pin this buildpack version for all applications running in your SAP BTP, Cloud Foundry environment subaccount, run:

```
cf push -b https://github.com/cloudfoundry/python-buildpack.git#v1.8.4
```

## Specify application memory in `manifest.yml`

When deploying an application in the SAP BTP, Cloud Foundry environment without specifying the application memory requirements, the Cloud Foundry controller assigns the default (1G of RAM currently) for your application. Many Python applications require less memory and assigning the default is a waste of resources.

To save memory from your quota, specify the memory size in the deployment descriptor of your Cloud Foundry application – `manifest.yml`. For example:

```
---
applications:
- name: myapp
  memory: 128M
```

```
buildpack: python_buildpack  
...
```

## Additional Tips

- The `cfnenv` package provides access to the Cloud Foundry application environment settings by parsing all the relevant environment variables. The settings are returned as a class instance. See: <https://github.com/jmcarp/py-cfenv> ↗
- When developing Python applications, it's a good practice to use virtual environments. The most popular Python package providing such a functionality is `virtualenv`. See: <https://virtualenv.pypa.io/en/stable/> ↗
- The [PEP 8 Style Guide for Python](#) ↗ will help you improve your applications.

### 4.1.3.6 Developing Multitenant Applications in the Cloud Foundry Environment

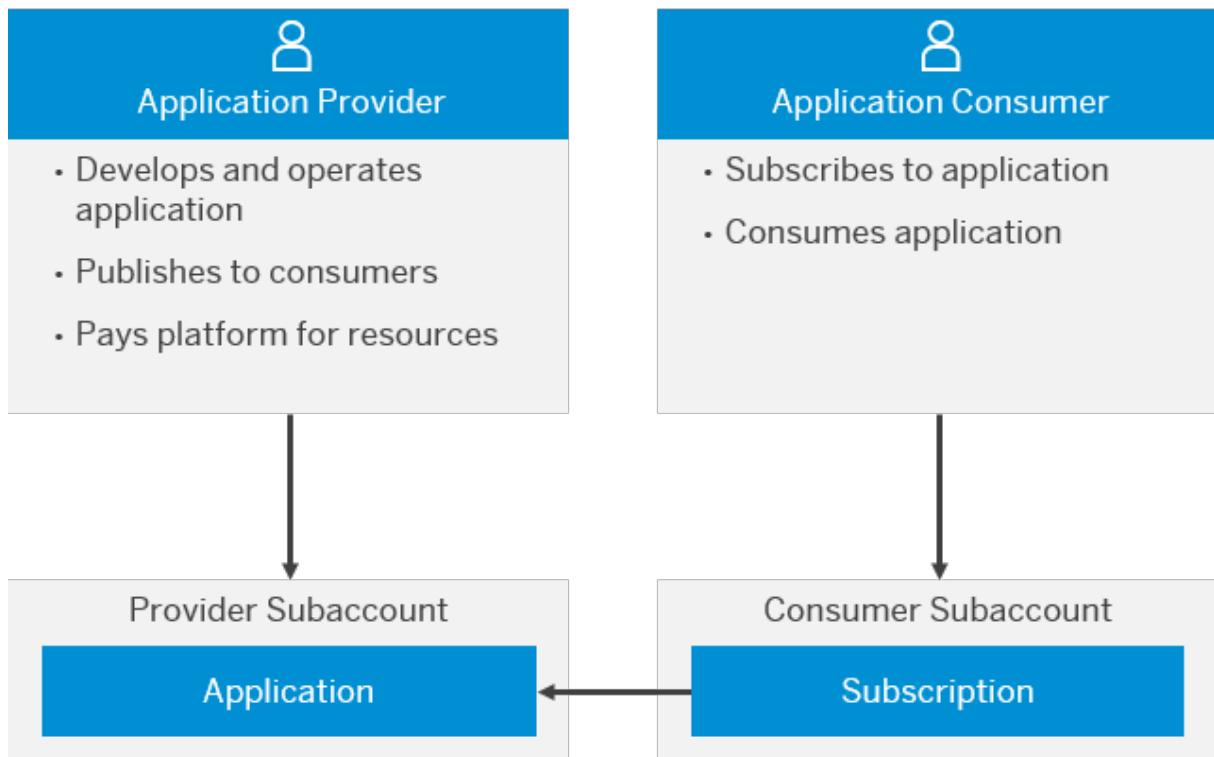
In the Cloud Foundry environment, you can develop and run multitenant applications, and share them with multiple consumers simultaneously on SAP BTP.

#### What is Multitenancy?

SAP BTP provides a multitenant functionality that allows application providers to own, deploy, and operate tenant-aware applications for multiple consumers, with reduced costs. For example, the application provider can upgrade the application for all your consumers instead of performing each update individually, or can share resources across multiple consumers. The application consumers launch the applications using consumer-specific URLs, and can configure certain application features.

With tenant-aware applications, you can:

- Separate data securely for each tenant
- Save resources by sharing them among tenants
- Update applications efficiently, in a single step



Consumers cannot see the data of other consumers; the multitenant application maintains data separation between tenants, and SAP Customer Identity and Access Management (SAP CIAM) is kept isolated. Each consumer accesses the multitenant application through a dedicated URL.

## Multitenancy Roles

The multitenancy concept involves two main user roles:

Application provider	An SAP global account owner that uses SAP BTP to own, build, run, and offer custom-developed applications to its consumers.
Application consumer	A consumer of the application provider, such as a department or organizational unit, whose users use the multitenant application.

## How Does Multitenancy Work for the Application Consumer?

For a consumer to use a tenant-aware application on SAP BTP, the application owner must ensure that each consumer:

- Has a dedicated subaccount in the application provider's global account.
- Subscribes to the application using either the SAP BTP cockpit, SAP BTP command-line interface, or a dedicated REST API.

A subscription means that there is a direct relationship between an application provider and the consumer's tenant. The application provider authorizes the consumer tenant to use the application.

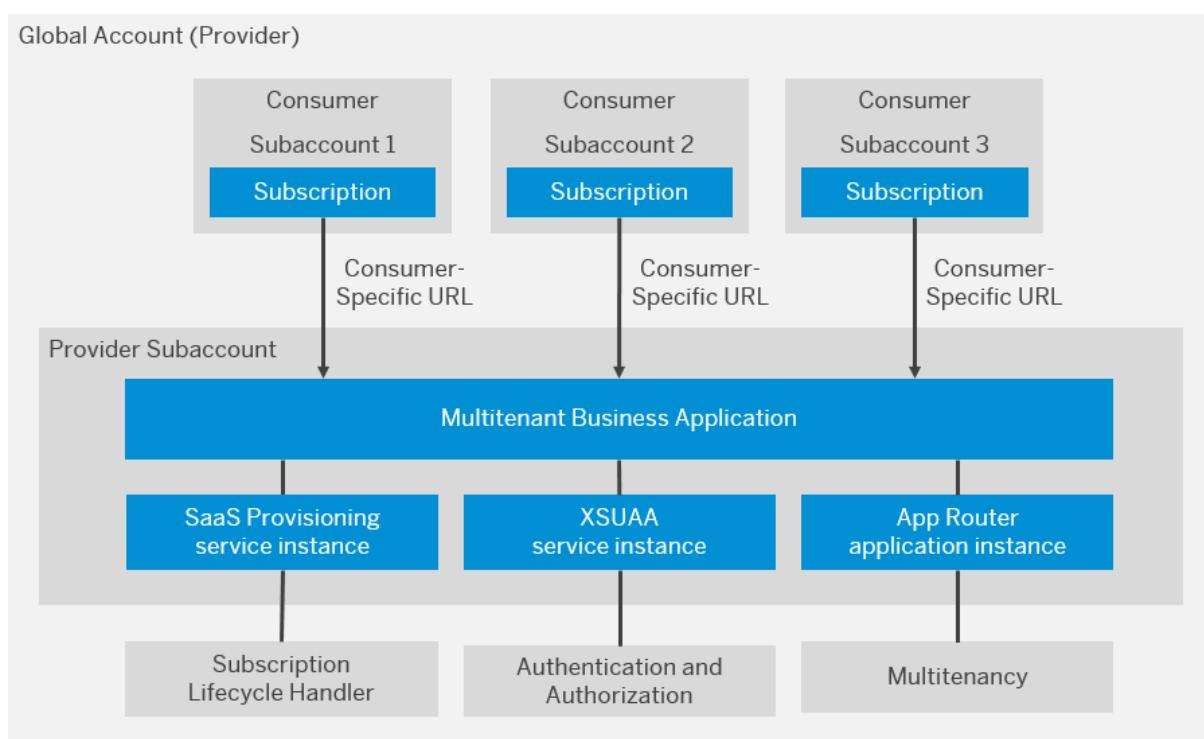
3. Receives a dedicated URL so that its business users can access the application

As with any application running in SAP BTP, these multitenant applications consume platform resources, such as compute units, structured and unstructured storage, and outgoing bandwidth. The costs for these consumed resources, and those of the application consumer, are billed to the provider of the multitenant application.

For more information about these consumer-related steps, see [Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 377\]](#) and [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#).

When a consumer accesses the application, the application environment identifies them by their unique tenant ID. The application distinguishes between requests from different consumer tenants based on the tenant ID, thus ensuring data isolation.

The following figure illustrates the relationship between the application provider's subaccount and consumer subaccounts (tenants) in the provider's global account. You deploy the multitenant application to the provider subaccount, and subsequently the consumer subaccounts subscribe to the deployed application. The application uses the tenant-aware approuter application and xsuaa service (with the application plan) to authenticate business users of the application at runtime. The application is then registered with the SAP Software-as-a-Service Provisioning service (technical name: saas-registry) with the application plan, which makes the application available for subscription to consumers.



## Workflow

This is the workflow and requirements for developing and deploying a multitenant application in the Cloud Foundry environment:

1. [Develop the Multitenant Application \[page 363\]](#)

In the Cloud Foundry environment of SAP BTP, you can develop and run multitenant applications that can be accessed by multiple consumers (tenants) through a dedicated URL.

2. [Deploy the Multitenant Application to the Provider Subaccount \[page 366\]](#)

After you have developed your multitenant application and authorizations, you need to deploy the application in a Cloud Foundry space in the provider's subaccount.

3. [Configure the approuter Application \[page 367\]](#)

To authenticate business users of the application at runtime, use the tenant-aware approuter application and xsuaa service in SAP BTP.

4. [Bind the Multitenant Application and Application Router to the xsuaa Service Instance \[page 369\]](#)

Bind your multitenant application and the approuter application to the xsuaa service instance, which acts as an OAuth 2.0 client to your application.

5. [Register the Multitenant Application to the SAP SaaS Provisioning Service \[page 370\]](#)

To make a multitenant application available for subscription to SaaS consumer tenants, you (the application provider) must register the application in the Cloud Foundry environment via the SAP Software-as-a-Service Provisioning service (saas-registry).

6. [Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 377\]](#)

Create a subaccount in your global account for each application consumer, subscribe each consumer subaccount to the hosted application deployed in the provider account, and then set up application roles and assign users.

## Related Information

[Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 379\]](#)

[Data Protection and Privacy \[page 3109\]](#)

[Using the Subscription Management Dashboard \[page 384\]](#)

## 4.1.3.6.1 Develop the Multitenant Application

In the Cloud Foundry environment of SAP BTP, you can develop and run multitenant applications that can be accessed by multiple consumers (tenants) through a dedicated URL.

### Context

When developing tenant-aware applications in the Cloud Foundry environment, keep in mind the following general programming guidelines:

- Shared in-memory data such as Java static fields will be available to all tenants.
- Avoid any possibility that an application user can execute custom code in the application JVM, as this may give them access to other tenants' data.
- Avoid any possibility that an application user can access a file system, as this may give them access to other tenants' data.
- To perform internal tenant onboarding activities, such as creating a database schema for tenants, you must implement the `Subscription` callbacks of the SAP Software-as-a-Service Provisioning service (`saas-registry`) and use the information provided in the subscription event. You can also implement the `getDependencies` callback to obtain the dependencies of any SAP reuse services by your application. See details in the procedure below.

### Procedure

1. Develop the application as for any other Java and Node.js application on the SAP BTP.  
For more information, see [Developing Java in the Cloud Foundry Environment \[page 261\]](#).
2. In your application, you must implement a REST API that will be called by the SAP SaaS Provisioning service (`saas-registry`) on a subscription event. Optionally, you can implement the `getDependencies` REST API to obtain the dependencies of any SAP reuse services consumed by your application.

When a consumer tenant creates or revokes a subscription to your multitenant application via the cockpit, the SAP SaaS Provisioning service calls the application with two `Subscription` callbacks. The first callback gets the dependencies of any reuse services used by your application. The second callback informs the application about the subscription event (`PUT` for subscribe and `DELETE` for unsubscribe) and provides details about the subscriber (the consumer tenant).

- To inform the application that a new consumer tenant wants to subscribe to the application, implement the callback service with the `PUT` method. The callback must return a 200 response code and a string in the body, which is the access URL of the tenant to the application subscription (the URL contains the subdomain). This URL is generated by the multitenant application based on the approuter configuration (see [Configure the approuter Application \[page 367\]](#))

#### ↔ Sample Code

Callback response (`PUT`).

```
https://subscriber_subdomain-
approuter_saas_app.cfapps.eu10.hana.ondemand.com
```

The URL must be in the format: `https://<SUBSCRIBER_TENANT_SUBDOMAIN>-<APPROUTER_APPLICATION_HOST>.<CF_DOMAIN>`

- To inform the application that a tenant has revoked its subscription to the multitenant application and is no longer allowed to use it, implement the same callback with the `DELETE` method.

#### ↔ Sample Code

Payload of subscription `PUT` and `DELETE` methods:

```
{  
    "subscriptionAppId": "<value>", # The  
    application ID of the main subscribed application. # Generated by  
    Authorization and Trust Management service(xsuaa)-based on the  
    xsappname.  
    "subscriptionAppName": "<value>" # The  
    application name of the main subscribed application.  
    "subscribedTenantId": "<value>" # ID of the  
    subscription tenant  
    "subscribedSubdomain": "<value>" # The subdomain  
    of the subscription tenant (hostname for  
    # the  
    identityzone).  
    "globalAccountGUID": "<value>" # ID of the  
    global account  
    "subscribedLicenseType": "<value>" # The license  
    type of the subscription tenant  
}
```

- If your application consumes any reuse services provided by SAP, you must implement the `getDependencies` callback to return the service dependencies of the application. The callback must return a 200 response code and a JSON file with the dependent services `appName` and `appId`, or just the `xsappname`.

#### ⓘ Note

The JSON response of the callback must be encoded as either UTF8, UTF16, or UTF32, otherwise an error is returned.

#### ↔ Sample Code

Sample dependency response:

```
[  
    {  
        "xsappname" : "<value>" # The xsappname of the reuse service  
        which the application consumes. # Can be found in the environment  
        variables of the application:  
        #  
        VCAP_SERVICES.<service>.credentials.xsappname  
    }  
]
```

For more information about the subscription callbacks and dependencies, see [Register the Multitenant Application to the SAP SaaS Provisioning Service \[page 370\]](#).

- Configure the application's authentication and authorization for the SAP HANA XS advanced Java run time.

For general instructions, see [SAP Authorization and Trust Management Service \[page 3023\]](#).

4. Create a security descriptor file in JSON format that specifies the functional authorization scopes for the application:

- Define the application provider tenant as a shared tenant:

```
"tenant-mode": "shared"
```

- Provide access to the SAP SaaS Provisioning service SAP Authorization and Trust Management service (technical name: saas-registry) for calling callbacks and getting the dependencies API by granting scopes:

```
"grant-as-authority-to-apps": [ "$XSAPPNAME(application,sap-provisioning,tenant-onboarding)"]
```

## ↔ Sample Code

Security descriptor file in JSON format:

```
{  
    "xsappname": "saas-app",  
    "tenant-mode": "shared",  
    "scopes": [  
        {  
            "name": "$XSAPPNAME.Callback",  
            "description": "With this scope set, the callbacks for subscribe,  
            unsubscribe and getDependencies can be called.",  
            "grant-as-authority-to-apps": [  
                "$XSAPPNAME(application,sap-provisioning,tenant-onboarding)"  
            ]  
        }  
        ....  
    ],  
    ....  
}
```

5. In the Cloud Foundry space in the provider's subaccount where your application is going to be deployed (see [Deploy the Multitenant Application to the Provider Subaccount \[page 366\]](#)), create an xsuaa service instance with the security configurations, which you defined in the security descriptor file in the previous step, by executing this command in the Cloud Foundry command line interface (cf CLI):

```
cf create-service xsuaa application <XSUAA_INSTANCE_NAME> -c  
<XS_SECURITY_JSON>
```

Specify the following parameters:

Parameter	Description
XSUAA_INSTANCE_N AME	The new name for the xsuaa service instance. Use only alphanumeric characters, hyphens, and underscores.
<XS_SECURITY_JSO N>	Name of the security descriptor file from the previous step.

#### ↔ Sample Code

```
cf create-service xsuaa application xsuaa-application -c xs-security.json
```

#### → Tip

You can also create the service instance directly in the cockpit.

For general instructions, see [Create Spaces Using the Cloud Foundry Command Line Interface \[page 2485\]](#).

## 4.1.3.6.2 Deploy the Multitenant Application to the Provider Subaccount

After you have developed your multitenant application and authorizations, you need to deploy the application in a Cloud Foundry space in the provider's subaccount.

### Context

To make the application available to subaccounts in multiple regions, you need to deploy the application to a provider subaccount in each of the regions.

### Procedure

1. Create a subaccount in the Cloud Foundry environment in the global account of the application provider. This subaccount will host the multitenant application for the provider tenant.  
For general instructions, see [Create a Subaccount \[page 2173\]](#).
2. In the subaccount, create a space in a Cloud Foundry organization.  
For general instructions, see [Create Spaces \[page 2441\]](#).
3. Assign the appropriate quota for the application runtime memory to the provider's subaccount.  
For general instructions, see [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#).
4. Deploy the application to the Cloud Foundry space and start it by executing the `push` command in the command line interface (cf CLI).  
For more information, see [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#).

### 4.1.3.6.3 Configure the approuter Application

To authenticate business users of the application at runtime, use the tenant-aware approuter application and SAP Authorization and Trust Management service (technical name: xsuaa) in SAP BTP.

#### Context

You deploy the approuter application as a Cloud Foundry application and as a logical part of the multitenant application. Then you configure approuter application as an external access point of the application. You need to deploy a separate application router for each multitenant application.

#### How does the application router and multitenancy work?

When a consumer accesses the application, their consumer tenant calls the multitenant application via the application router with their tenant-specific URL (cf route).

- **During a development phase** of your multitenant application, the following URL structure applies:  
`<SUBSCRIBER_TENANT_SUBDOMAIN>-<APPROUTER_APPLICATION_HOST>. <SAP- PROVIDED_STANDARD_DOMAIN>`  
In each development landscape, SAP provides a different standard domain you can use to create URLs. Using that domain, and following the specified URL structure, a created URL would be:  
`tenant1-myapprouter.cfapps.sap.hana.ondemand.com`  
Where: tenant1-myapprouter is the hostname, and cfapps.sap.hana.ondemand.com is the SAP-provided standard domain for this development landscape.  
In this format, create a new URL for each new tenant.
- **During a production phase** of your multitenant application, apply for a custom domain via the custom domain service, then create a URL using the following structure:  
`*.<YOUR_CUSTOM_DOMAIN>.`  
Where: \* is the wildcard hostname.  
In this format, since the wildcard is replaced by an actual tenant during runtime, there is no need to create a new URL for each new tenant.  
For more information, see [Using Custom Domains \[page 2501\]](#)

In both cases, the application router then derives the tenant from the URL and calls the tenant-aware xsuaa (containing the user account and authentication service), which is responsible for the authentication of the business user. The xsuaa reads the tenant and gets the customer-specific identity provider (IdP) from the tenant-specific identity zone. Then, the xsuaa delegates the authentication to the configured IdP, and creates a JSON Web Token (JWT) that contains the tenant, the current user, and the authorization scopes of the user. The JWT is then sent back to the application router, and from there to the application.

To read and store tenant-specific data, the multitenant application needs to know the tenant ID. To read the tenant ID, use the Container Security API to enable the multitenant application to read the tenant, user, and scopes from the given JWT. The API also validates whether the JWT is valid. The tenant information is contained in the `identityZone` user attribute.

#### Sample Code

Java code for retrieving the identity zone

```
UserInfo userInfo = SecurityContext.getUserInfo();
```

```
String identityZone = userInfo.getIdentityZone();
```

For more information, see:

- [Web Access Control \[page 3034\]](#)
- [What Is the SAP Authorization and Trust Management Service? \[page 3030\]](#)

## Procedure

1. Create the application structure of the application router and configure it accordingly.

For general instructions, see [Application Router \[page 409\]](#).

2. Configure the application router with the destination of your multitenant application.

If you are defining the destination as an environment variable for the `approuter` application, set the router information in the `env: destinations` section of the `manifest.yml`.

- `url`: Specify the URL of the multitenant application.
- `forwardAuthToken`: Set to `true`

### ↔ Sample Code

`manifest.yml` for a development phase:

```
---
applications:
- name: approuter-saas-app
  host: approuter_saas_app
  path: approuter
  buildpack: nodejs_buildpack
  memory: 256M
  env:
    TENANT_HOST_PATTERN: "^(.*)-cfapps.eu10.hana.ondemand.com"
  destinations: >
    [
      {"name": "saas-application",
       "url": "https://backend-saas-app.cfapps.sap.hana.ondemand.com",
       "forwardAuthToken": true}
    ]
```

### ↔ Sample Code

`manifest.yml` for a production phase:

```
---
applications:
- name: approuter-saas-app
  host: approuter_saas_app
  path: approuter
  buildpack: nodejs_buildpack
  memory: 256M
  env:
    TENANT_HOST_PATTERN: "^(.).mydomain.com"
  destinations: >
    [
      {"name": "saas-application",
       "url": "https://backend-saas-app.mydomain.com",
```

```
        "forwardAuthToken": true}
    ]
```

For more information, see [Application Routes and Destinations \[page 450\]](#).

3. Configure the routes in the application router security descriptor file (`xs-app.json`) so that application requests are forwarded to the multitenant application destination.

#### ↔ Sample Code

`xs-app.json`:

```
{
  "routes": [
    {
      "source": "/",
      "target": "/",
      "destination": "saas-application"
    }
  ]
}
```

For more information, see [Routing Configuration File \[page 420\]](#).

4. Use the `push` command in the `cf` CLI to deploy the `approuter` application to the Cloud Foundry space where your multitenant is deployed.

### 4.1.3.6.4 Bind the Multitenant Application and Application Router to the `xsuaa` Service Instance

Bind your multitenant application and the `approuter` application to the SAP Authorization and Trust Management service (technical name: `xsuaa`) instance, which acts as an OAuth 2.0 client to your application.

#### Procedure

Execute the following commands in the Cloud Foundry command line interface to bind both your multitenant application and the `approuter` application to the `xsuaa` service instance.

```
cf bind-service <APP_NAME> <XSUAA_INSTANCE_NAME>
```

```
cf bind-service <APPROUTER_APP_NAME> <XSUAA_INSTANCE_NAME>
```

Specify the following parameters:

Parameter	Description
<code>APP_NAME</code>	The name of the deployed multitenant application.
<code>XSUAA_INSTANCE_NAME</code>	The name of the <code>xsuaa</code> service instance you created in <a href="#">Develop the Multitenant Application [page 363]</a> .

Parameter	Description
APPROUTER_APP_NAME	The name of the xsuaa application you deployed in <a href="#">Configure the approuter Application [page 367]</a> .

#### ↳ Sample Code

```
cf bind-service saas-app xsuaa-application  
cf bind-service approuter-saas-app xsuaa-application
```

#### → Tip

You can also bind the application to the xsuaa service instances directly in the SAP BTP cockpit.

### 4.1.3.6.5 Register the Multitenant Application to the SAP SaaS Provisioning Service

To make a multitenant application available for subscription to SaaS consumer tenants, you (the application provider) must register the application in the Cloud Foundry environment via the SaaS Provisioning Service (technical name: `saas-registry`).

#### Context

The SaaS Provisioning Service allows application providers to register multitenant applications and services in the Cloud Foundry environment in SAP BTP. This activity is a one-time procedure per multitenant application.

#### Procedure

1. Create a service instance of the SaaS Provisioning Service with a configuration JSON file in the Cloud Foundry space where the multitenant application is deployed.

#### → Tip

You can create the SaaS Provisioning Service instance directly in the SAP BTP cockpit.

For general instructions, see [Creating Service Instances](#).

- a. The configuration JSON file must follow the following format and set these properties (note that instead of `xsappname` you can use `appId`):

```
{  
  "xsappname" : "<xsappname>,"
```

```

    "appUrls": {
        "getDependencies" : "<getDependenciesUrl>",
        "onSubscription" : "<onSubscriptionUrl>/{{tenantId}}"
    },
    "appPlans": [
        {
            "name": "...",
            "description": "..."
        }, {
            ...
        }
    ],
    "displayName" : "<displayName>",
    "description" : "<description>",
    "category" : "<category>",
    "onSubscriptionAsync": true/false,
    "callbackTimeoutMillis": <int>
    "allowContextUpdates": true/false,
}

```

Specify the following parameters:

Parameters	Description
<code>xsapppname</code>	The <code>xsapppname</code> configured in the security descriptor file used to create the <code>xsuaa</code> instance (see <a href="#">Develop the Multitenant Application [page 363]</a> ).

Parameters	Description
getDependencies	<p>(Optional) Implement a callback API that is executed by the SaaS Provisioning Service to retrieve your application's dependencies.</p> <p><b>① Note</b></p> <p>Implement only if your application has dependencies.</p> <p><b>API Structure</b></p> <ul style="list-style-type: none"> <li>Request: <ul style="list-style-type: none"> <li>GET</li> <li>URL: Define the endpoint to which SaaS Provisioning Service executes the request. For example: <code>https://&lt;app url&gt;/callback/v1.0/dependencies</code></li> </ul> </li> <li>Response: <ul style="list-style-type: none"> <li>HTTP Status Code: 200 (Ok)</li> <li>Header: <code>content-type: application/json</code></li> <li>Response Body: Specify the <code>xsappname</code> of each of your dependencies.</li> </ul> </li> </ul> <pre>{   "xsappname": "...", # The   xsappname of the reusable service instance   that the application consumes.   Find it   in the following environment variable of   your application: &lt;VCAP_SERVICES.&lt;service&gt;.credentials.xsappn ame &gt; }, ....</pre> <p><b>① Note</b></p> <p>The JSON response of the callback must be encoded as either UTF8, UTF16, or UTF32, otherwise an error is returned.</p>
onSubscription	Any URL that the application exposes via PUT and DELETE subscription. It must end with <code>/ {tenantId}</code> . The tenant for the subscription is passed to this callback as a path parameter. You must keep <code>{tenantId}</code> as a parameter in the URL so that it's replaced at real time with the tenant calling the subscription. This callback URL is called when a subscription between a multitenant application and a consumer tenant is created (PUT) and when the subscription is removed (DELETE).
appPlans	(Optional) Specify the plan for your application. Name it and provide a short description. With the help of plans, you can try out your application in different modes or variations. For example, an app for staging or development purposes. Note that the plans are only visible in the subaccounts that belong to the provider global account.

Parameters	Description
displayName	(Optional) The display name of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's technical name.
description	(Optional) The description of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's display name.
category	(Optional) The category to which the application is grouped in the <i>Subscriptions</i> page in the cockpit. If left empty, gets assigned to the default category.
onSubscriptionAsync	Whether the subscription callback is asynchronous.  If set to true, <code>callbackTimeoutMillis</code> is mandatory.
callbackTimeoutMillis	The number of milliseconds the SaaS Provisioning Service waits for the application's subscription asynchronous callback to execute, before it changes the subscription status to FAILED.
allowContextUpdates	Whether to send updates about the changes in contextual data for the service instance.  For example, when a subaccount with which the instance is associated is moved to a different global account.  Default value is false.

### ↔ Sample Code

Configuration JSON file (example: config.json):

```
{
  "xsappname": "saas-app",
  "appUrls": {
    "getDependencies" : "https://saas-
app.cfapps.eu10.hana.ondemand.com/callback/v1.0/dependencies",
    "onSubscription" : "https://saas-
app.cfapps.eu10.hana.ondemand.com/callback/v1.0/tenants/{tenantId}"
  },
  "displayName" : "Hello World",
  "description" : "My multitenant biz app",
  "category" : "Test"
}
```

- Execute the following cf CLI command to create the service instance with the JSON configuration file:

```
cf create-service saas-registry application
<SAAS_REGISTRY_SERVICE_INSTANCE> -c <JSON_CONFIG_FILE>
```

Specify the following parameters:

Parameters	Description
SAAS_REGISTRY_SERVICE_INSTANCE	The new name for your service instance. Use only alphanumeric characters, hyphens, and underscores.

Parameters	Description
JSON_CONFIG_FILE	The file name of the service-specific configuration parameters, in a valid JSON object (described above).

#### ↔ Sample Code

```
cf create-service saas-registry application saas-registry-application
-c config.json
```

2. To ensure that the application trusts the JWT issued for another identity zone (`sap-provisioning`), add the environment variable: `SAP_JWT_TRUST_ACL`

```
SAP_JWT_TRUST_ACL: '[ {"clientid": "*", "identityzone": "sap-provisioning" } ]'
```

#### ⓘ Note

The client libraries (java-security, spring-xsuaa, container security api for node.js >= 3.0.6, approuter >= 8.x, and sap\_xssec for Python >= 2.1.0) have been updated. When using these libraries, setting the parameter `SAP_JWT_TRUST_ACL` has become obsolete.

Instead, the `xsuaa` service adds audiences to the issued JSON Web Token audience field (`aud`). Client libraries provide an audience validator to check whether the token is issued for your service or application. Technically, this information is implicitly derived from the audience field (`aud`) or from the scopes.

Therefore, this update comes with a change regarding scopes. For a business application A that wants to call an application B, it's now mandatory that application B grants at least one scope to the calling business application A. Furthermore, business application A has to accept these granted scopes or authorities as part of the application security descriptor. You can grant scopes with the `xs-security.json` file.

For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

3. Bind the SaaS Provisioning Service instance that you created to the multitenant application that you deployed in your Cloud Foundry space using the following cf CLI command:

```
cf bind-service <APP_NAME> <SAAS_REGISTRY_SERVICE_INSTANCE>
```

Specify the following parameters:

Parameters	Description
APP_NAME	The ID of your deployed multitenant application.
SAAS_REGISTRY_SERVICE_INSTANCE	The name of the SaaS Provisioning Service instance you created in step 1 above.

#### ↔ Sample Code

```
cf bind-service saas-app saas-registry-application
```

→ Tip

You can bind your application to the SaaS Provisioning Service instance directly in the SAP BTP cockpit.

For general instructions, see [Binding Service Instances to Applications \[page 253\]](#).

4. To ensure that the environment variables in the application take effect, execute the following cf CLI command:

```
cf restage <APP_NAME>
```

Specify the following parameter:

Parameters	Description
APP_NAME	The ID of your deployed multitenant application.

↔ Sample Code

```
cf restage saas-app
```

## Next Steps

1. In your global account, create a subaccount for each consumer tenant.
2. Subscribe your consumer tenants to the multitenant applications using the cockpit.
3. Test your multitenant application from consumer tenants to ensure availability, subscription lifecycle, and optional application configurations.
4. Supply your customers or users of the multitenant application with their consumer-specific URL to access the application.

For more information, see [Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 377\]](#).

## 4.1.3.6.5.1 Unregister a Multitenant Application from the SAP SaaS Provisioning Service

Unbind and delete the service instance you have created during the registration of your multitenant application to complete the unregistration process.

### Prerequisites

Make sure that no tenant is subscribed to your application before you initiate the deletion of the service instance.

→ Tip

Use the `Get Application Subscriptions` API to get the application subscriptions.

In case there are subscribed tenants, unsubscribe them first by calling the `Unsubscribe the tenant from an application` API.

For more information, see [Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 379\]](#).

### Context

The procedure described in this section uses the cf CLI.

→ Tip

You can delete a service instance directly in the SAP BTP cockpit, in the `Service Instances` page.

### Procedure

1. Use cf CLI to remove any existing service keys and app bindings:
  - a. Run the following command to delete the service key:

```
cf delete-service-key SERVICE-INSTANCE-NAME KEY-NAME
```

ⓘ Note

Service key is an optional parameter.

In case you haven't created a service key during the registration process, skip to substep b.

- b. Run the following command to unbind your application from the service instance:

```
cf unbind-service APP-NAME SERVICE-INSTANCE-NAME
```

- c. Run the following command to delete the service instance:

```
cf delete-service SERVICE-INSTANCE-NAME
```

2. The deletion process is asynchronous. Run `cf services` to view the current status of the process.

#### **4.1.3.6.6 Providing Multitenant Applications to Consumers in the Cloud Foundry Environment**

Once you have built a multitenant application in the Cloud Foundry environment using SAP BTP, you can then share the application with multiple consumers, such as business units in your organization.

##### **Consumer Subaccounts and Subaccount Members**

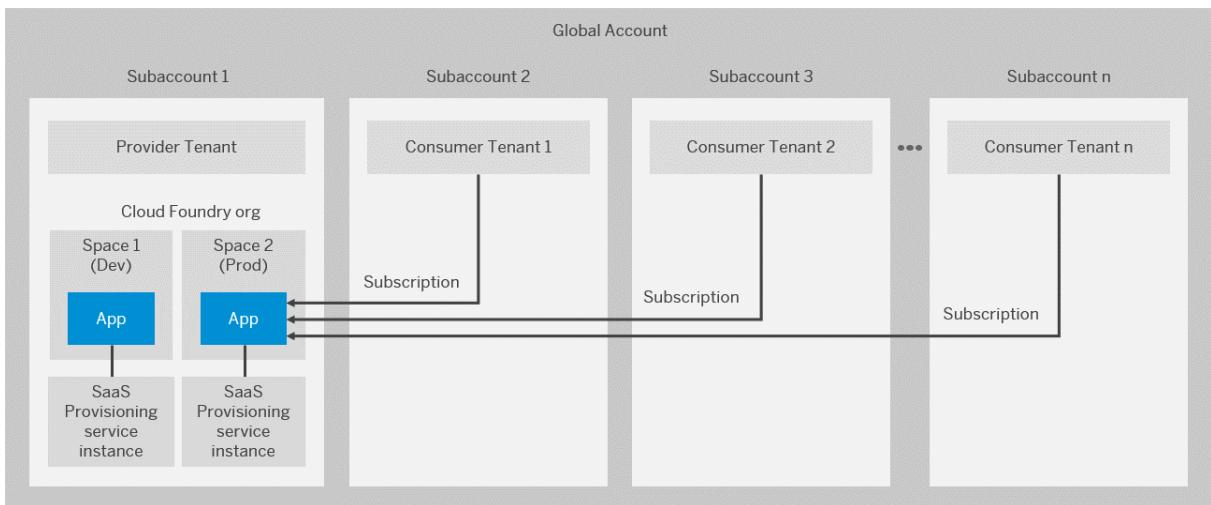
To use multitenancy in the Cloud Foundry environment in SAP BTP, the application provider must create a subaccount for each consumer in the same global account where the multitenant application is deployed.

The multitenant application is deployed to one subaccount only, which is the hosting subaccount of the application provider. Consumers simply subscribe to the provider application from their subaccount through the SAP BTP cockpit, the SAP BTP command line interface, or dedicated REST APIs.

The application provider is responsible for:

- Creating and managing the tenant subaccounts.
- Setting up the necessary role access authorizations.
- Subscribing each tenant to the application.

Typically, consumers of multitenant applications provided by a non-SAP application owner, such as an IT department of an organization, do not own or have access to their tenant subaccount.



Account setup for an application owner provisioning a multitenant application to its consumers.

Subaccount members are users who are registered via the SAP ID service. Subaccount members may have different privileges regarding the operations that are possible for a subaccount (for example, subaccount administration, deploy, start, and stop applications).

The subaccount-specific configuration allows application providers to adapt the consumer subaccounts to their specific needs.

## What Do You Need to Do, and How?

Once you have developed, deployed, configured, and registered your multitenant application, your account administrator is ready to do the following.

1. Create a subaccount in your global account for each application consumer.  
See [Create a Subaccount \[page 2173\]](#).
2. Subscribe each consumer subaccount to the hosted application deployed in the provider account.  
See:
  - [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#)
  - [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
  - Subscribe the tenant to an application using the REST API. For more information, see [Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 379\]](#)
3. Set up application roles and assign users.  
See [Configure Application Roles and Assign Roles to Users \[page 2201\]](#).

## Viewing and Managing of Your Multitenant Application's Subscriptions

Once you've provided your multitenant application to consumers to subscribe to it, you can view and manage all the subscriptions by using a dedicated interface called SAP BTP subscription management dashboard.

For more information, see [Using the Subscription Management Dashboard \[page 384\]](#).

## 4.1.3.6.7 Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications

Use the SaaS Provisioning Service (technical name: `saas-registry`) APIs to work with multitenant applications.

### Background

Most of the SaaS Provisioning Service APIs are concentrated into two main groups:

1. APIs for Application Providers
2. APIs for Application Consumers

Application providers can use both groups to either manage their application subscriptions from the global account in which they deployed their multitenant application when using the first group, or mimic their consumers in working with the application from a different global account than the one in which their application is deployed, when using the second group.

Consumers can use the second group to work with the multitenant applications.

All SaaS Provisioning Service APIs adhere to the rate limiting rules as described in [Rate Limiting \[page 2408\]](#).

### APIs for Application Providers

#### → Tip

Instead of APIs, can use a dedicated user interface with the same capabilities to manage your multitenant application's subscriptions.

For more information, see [Using the Subscription Management Dashboard \[page 384\]](#).

#### Prerequisites

- You have a global account and have deployed a multitenant application in it.
- You've registered your multitenant application to the SaaS Provisioning Service. For more information, see [Register the Multitenant Application to the SAP SaaS Provisioning Service](#).

#### Preparation

1. For an existing instance of `saas-registry` service with the application plan, create a service key or service binding if you don't already have one. For more information, see [Service Bindings](#).
2. From the created key or service binding, extract the values of the following fields:
  - `url`
  - `clientid`
  - `clientsecret`
  - `saas_registry_url`

You need the first three values to authenticate your SaaS Provisioning Service API requests and the last one as base URL in each API call.

## Authentication

To authenticate, obtain an OAuth token by calling a POST API with the acquired parameters in any REST API tool or directly in your code. For example:

```
curl -XPOST '<url>/oauth/token?  
grant_type=client_credentials&client_id=<clientid>' -H 'Content-Type:  
application/x-www-form-urlencoded' -H 'Authorization: Basic <encodedString>'
```

where:

<encodedString> is the result of base64 encoding <clientid>:<clientsecret>.

## Available APIs

Go to [SAP Business Accelerator Hub](#) and select the **Application Operations for App Providers** section to get information about all the available APIs and their details.

You can also try out the APIs in the SAP Business Accelerator Hub.

For more information, see [Trying Out APIs](#).

### ⓘ Note

To authenticate to try out the APIs in SAP Business Accelerator Hub, configure an environment.

During the environment configuration, select the [OAuth 2.0 Application Flow](#) option from the dropdown menu. You must do so for the Application Provider API group because these specific APIs only support the Client Credentials authentication type.

For more information about configuring an environment in general, see [Trying Out APIs in a Configured Environment](#).

## APIs for Application Consumers

### Prerequisites

- You've created an instance of Cloud Management (technical name: cis) service with local plan.

## Authentication

To learn how to authenticate, see [Getting an Access Token for SAP Cloud Management Service APIs](#).

## Available APIs

Go to [SAP Business Accelerator Hub](#) and select the **Subscription Operations for App Consumers** section to get information about all the available APIs and their details.

You can also try out the APIs in the SAP Business Accelerator Hub. For more information, see [Trying Out APIs](#).

To authenticate to try out the APIs in SAP Business Accelerator Hub, configure an environment.

For more information about configuring an environment in general, see [Trying Out APIs in a Configured Environment](#).

### 4.1.3.6.7.1 Getting an Application Access Token

Use this API to get the application access token from the SAP Software-as-a-Service Provisioning service instance.

#### Prerequisites

You've registered your multitenant application to the SAP SaaS Provisioning service.

For more information, see [Register the Multitenant Application to the SAP SaaS Provisioning Service \[page 370\]](#).

#### Obtaining API Request Parameters

To get the token, you call the API with the parameters obtained from the service binding object you created during the registration of your multitenant application to the SAP SaaS Provisioning service.

Use the following cf CLI command to get the service binding object:

```
cf env <APP_NAME>
```

See the step 3 in the [Register the Multitenant Application to the SAP SaaS Provisioning Service \[page 370\]](#) to find the `<APP_NAME>`.

The example of the binding object you get after executing the cf CLI command, with the needed values for the API (`clientid`, `clientsecret`, and `url`) marked in bold:

##### Sample Code

```
"saas-registry": [
{
  "credentials": {
    "apiurl": "https://api.authenticat*****avlab.ondemand.com",
    "appName": "sample-saas-ap*****45",
    "appUrls": "{\"getDependencies\": \"http*****.cf.stag**\":0}",
    "clientid": "sb-sample-saas-*****-broker!b4",
    "clientsecret": "riH*****0=",
    "description": "Sample multitenant application",
    "display_name": "Sample multitenant application",
    "identityzone": "cfs*****44",
    "saas_registry_url": "https://saa*****ab.ondemand.com",
    "sburl": "https://internal-xsu*****ndemand.com",
    "subaccountid": "3358efc9-*****10b456",
    "tenantid": "34584*****711",
    "url": "https://cfs-3035-*****avlab.ondemand.com",
    "xsappname": "sample-saa*****istry-broker!b4",
  }
}
```

```

    "zoneid": "34584*****f499711"
},
"instance_name": "saas-app-saas-registry",
"label": "saas-registry",
"name": "saas-app-saas-registry",
"plan": "application",
}
]

```

The token you get after executing the API is a JSON Web Token (JWT).

For more information, see [JSON Web Token \(JWT\)](#).

You use this token to manage the SAP SaaS Provisioning service APIs.

## Request

**URI:** <THE URL OBTAINED FROM THE BINDING OBJECT "URL" FIELD>/oauth/token

**HTTP Method:** *POST*

### Request Headers

Header	Required	Values
Content-Type	Yes	< <a href="#">application/x-www-form-urlencoded</a> >
Authorization	Yes	Basic < <a href="#">encodedString</a> > where < <a href="#">encodedString</a> > is the result of base64 encoding the OAuth client's values as <code>clientId:clientSecret</code> that you obtained from the binding object as described in the previous section.  For more information about the base64 encoding, see <a href="#">Base64</a> .

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
grant_type	Yes	String	The type of the authorization that is supported by the authorization server.  Set it to <code>client_credentials</code> .	Authorization protocol
client_id	No	String	The ID of the client associated with the relative URL path or JavaScript source code	Relative URL path or JavaScript source code

Parameter	Required	Data Type	Description	Parameter Type
			SaaS Provisioning service instance. Obtained from the service binding object. See the section Obtaining API Request Parameters of this document for details.	

## Request Example (curl for Mac OS)

### ↔ Sample Code

```
curl --location --request POST '<url>/oauth/token' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: Basic <base64.encoded(client_id:client_secret)>' \
--data-urlencode 'client_id=<client_id>' \
--data-urlencode 'grant_type=client_credentials'
```

## Request Example (curl for Windows OS)

### ↔ Sample Code

```
curl --location --request POST "<url>/oauth/token" ^
--header "Content-Type: application/x-www-form-urlencoded" ^
--header "Authorization: Basic <base64.encoded(client_id:client_secret)>" ^
--data-urlencode "client_id=<client_id>" ^
--data-urlencode "grant_type=client_credentials"
```

## Response

Generates the access token for a multitenant application.

**Content Type:** [JSON](#)

### Response Headers

Header	Values
Content-Type	<application/json; charset=UTF-8>

### Response Status and Error Codes

Code	Description
200	Access token created successfully.

## Response Properties

Property Name	Property Type	Description
access_token	JWT	Access token for the multitenant application.
		This is the value for which you call the API
token_type	String	The type of access token issued.
expires_in	Number	The number of seconds until the access token expires.
scope	String	A space-delimited list of scopes that you authorized for the client.
jti	String	A globally unique identifier for JWT.

## Response Example

### ↔ Sample Code

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
{
  "access_token": "eyJ***mh0d..." ,
  "token_type": "bearer" ,
  "expires_in": 43199 ,
  "scope": "uaa.re***.subscription.read" ,
  "jti": "df6cb84439a541fab33d5b7c298debe1"
}
```

## Related Information

[Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 379\]](#)

## 4.1.3.6.8 Using the Subscription Management Dashboard

Learn how to use the SAP BTP subscription management dashboard to manage your multitenant applications through a user interface.

## Context

SaaS and IAS application providers can now use the new subscription management dashboard to manage their multitenant application's subscriptions.

The tool is based on the SaaS Provisioning service APIs that you can now work with using a user interface to simplify your operation and maintenance work.

## Main Capabilities

You can do the following in the subscription management dashboard:

- View all subscriptions to your application and the information about them, such as their state and tenant ID.
- View each subscription's dependencies and status.
- Perform operations on subscriptions, for example update the list of dependencies.
- Control access to the dashboard using dedicated roles.

The dashboard also displays detailed information about any errors that arise during the subscription process. This gives you means to identify the source of issues and fix them.

## Accessing the Dashboard in the SAP BTP Cockpit

### Assign Roles

To access the subscription management dashboard, you must be assigned to one of these roles:

- **Subscription Management Dashboard Administrator** - Provides access to the dashboard with full read and write capabilities.
- **Subscription Management Dashboard Viewer** - Provides read-only access to the dashboard.

Add the relevant role to a role collection and assign yourself or another user to it.

#### ⓘ Note

You assign roles per tenants. To assign roles in a global account or subaccount tenant, you must be a global account or subaccount admin respectively.

For more information about roles, see [Configure Application Roles and Assign Roles to Users](#).

#### ⓘ Note

If you don't see the roles, you need to update the instance of the SaaS Provisioning service (technical name: `saas-registry`) associated with your application (`application plan`). This enables the new roles.

### Open the Dashboard

1. In the left-hand navigation bar of the SAP BTP cockpit, select **Services** **Instances and Subscriptions**

#### ⓘ Note

Only users who are assigned to one of the roles specified in the previous section can see the link to the dashboard and open it.

- In the *Instances* section, find the instance of the SaaS Provisioning service (technical name: `saas-registry`) with the application plan and click on it.  
Details about the instance open to the right.
- In the top-right corner, select **...** and from the dropdown menu, choose *View Dashboard*.  
A new tab with the subscription management dashboard opens.

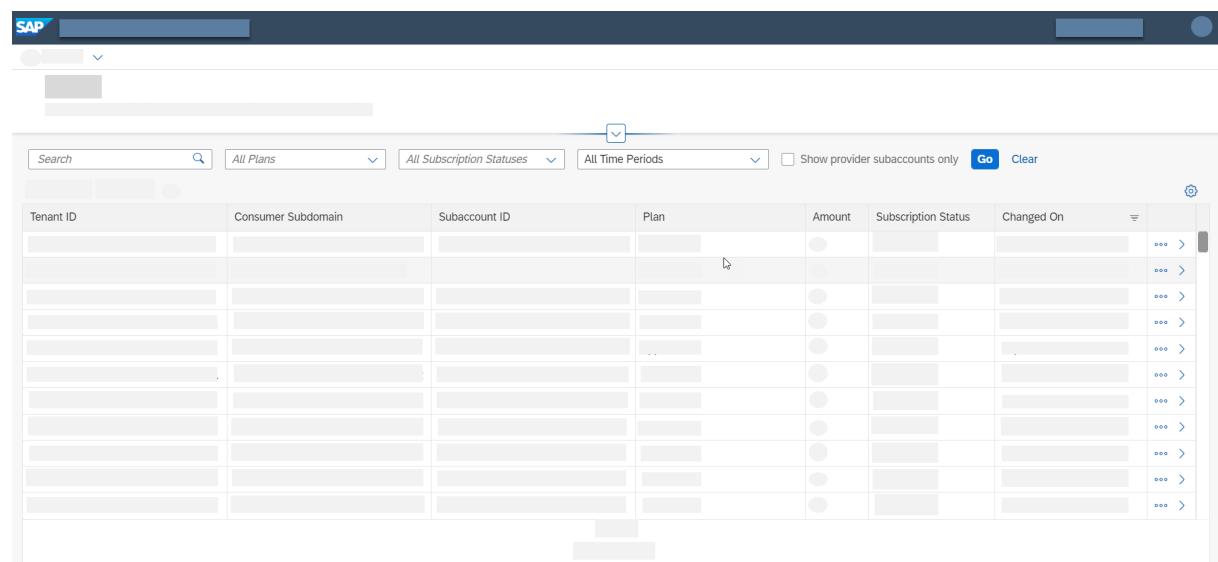
## Using the Dashboard

The main screen shows a table with details about all of your application's subscriptions.

You can apply various search and filtering criteria such as plan, subscription status, and time periods to get a more specific list of subscriptions.

**→ Tip**

Select the *Show provider subaccounts only* checkbox to see only the subaccounts that are in the global account in which your application is deployed. This can be useful because you can only create new subscriptions for these subaccounts.



The screenshot shows a SAP web interface for managing subscriptions. At the top, there is a dark header bar with the SAP logo. Below it is a navigation bar with links like 'All Plans', 'All Subscription Statuses', 'All Time Periods', and a search bar. The main area is a table with columns: Tenant ID, Consumer Subdomain, Subaccount ID, Plan, Amount, Subscription Status, and Changed On. Each row in the table has a 'More' button (three dots) at the end. The table is scrollable, indicated by vertical and horizontal scroll bars.

Clicking on a subscription row opens details about the subscription to the right where you can find more information about the subscription and analyze errors if the subscription status is `Failed`.

You can also perform additional operations, such as update subscription dependencies, if you're assigned to the `Subscription Management Dashboard Administrator` role.

This operation updates the subscription with the latest list of dependencies (if an app provider has added or removed dependencies of a particular subscription, for example.)

The screenshot shows a SAP BTP Cloud Foundry dashboard. On the left, there is a search bar with 'Search' and 'Go' buttons, and a table titled 'Subscription Status' with columns 'Tenant ID' and 'Subscription Status'. Most rows in the table have a status of 'Subscribed'. On the right, there is a detailed view for a specific subscription. The top part shows 'Tenant ID' and 'Consumer Subdomain'. Below that is a tabbed section with 'Overview' and 'Dependencies'. The 'Overview' tab is active, displaying fields like 'Global Account ID', 'Amount', 'Consumer Subdomain', 'Plan', 'App Name', 'URL', 'Changed On', 'Created On', and 'isConsumerTenantActive'. The 'Dependencies' tab is also visible.

You can choose to see your subscription dependencies in a tree or table view. See the following image for the tree view representation:

This screenshot shows the same SAP BTP Cloud Foundry interface but with the 'Dependencies' tab selected. The main area displays a tree diagram where a 'Root Subscription' (status: SUBSCRIBED) has three child nodes, each represented by a green rectangular box with a smaller tree icon next to it, indicating they are also root subscriptions.

#### → Tip

If you've more than one multitenant application, you can switch between them in the dashboard. Click on the dropdown menu in the upper-left corner and select the application for which you want to view or manage subscriptions.

## 4.1.4 Developing User Interface

Get information on user interface development in the SAP BTP, Cloud Foundry environment.

The key to a modern user interface is pairing great performance with a consistent and recognizable design across applications, platforms, and devices. To keep the user experience homogenous, you can establish a

flexible implementation paradigm based on reusable components. Doing so enables extensibility and easier maintainability.

We know that it can be difficult to decide this up front. You can find information about UI development options in the SAP BTP, Cloud Foundry environment in the following sections.

## Fiori Design Guidelines

The [SAP Fiori Design Guidelines](#) define design solutions that are easy to use, robust, scalable, and of enterprise-grade quality. The guidelines are independent of the implementation technology.

The modular design approach focuses on user tasks and workflows. It enables seamless transitions between applications and offers clear guidance for extension and integration design. To ensure a consistent user experience, we recommend following the Fiori Design Guidelines independent of the UI technology you plan on using.

## SAPUI5

SAPUI5 is a JavaScript framework for building cross-platform, enterprise-grade web apps with modern, responsive, and rich user interfaces. For more information, see [Developing SAPUI5 \[page 388\]](#).

## HTML5

On SAP BTP, you can access and run HTML5 applications in a cloud environment without the need to maintain your own runtime infrastructure. For more information, see [Developing HTML5 Applications in the Cloud Foundry Environment \[page 391\]](#).

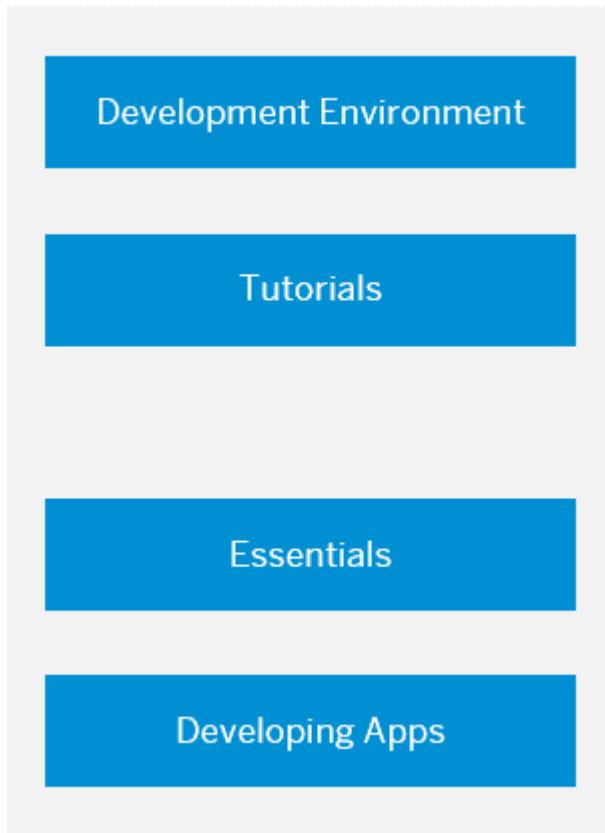
### 4.1.4.1 Developing SAPUI5

Get to know certain aspects of SAPUI5 development, to get up and running quickly.

If you are about to decide which UI technology to use , read everything you need to know about supported library combinations, the supported browsers and platforms, and so on, at the [Read Me First](#) section that contains the following topics and more:

- [Supported Library Combinations](#)
- [Supported Combinations of Themes and Libraries](#)
- [Browser and Platform Support](#)

## Quick Start



- [#unique\\_268/unique\\_268\\_Connect\\_42\\_subsection-im1 \[page 389\]](#)
- [#unique\\_268/unique\\_268\\_Connect\\_42\\_subsection-im2 \[page 390\]](#)
- [#unique\\_268/unique\\_268\\_Connect\\_42\\_subsection-im3 \[page 390\]](#)
- [#unique\\_268/unique\\_268\\_Connect\\_42\\_subsection-im4 \[page 390\]](#)

Select the tiles to discover SAPUI5 Development. The references guide you to the documentation of the SAPUI5 Demo Kit. Besides the entry points we provide here on each tile, start exploring the demo kit on your own whenever you feel comfortable enough.

### SAP Business Application Studio

Use the SAP Business Application Studio to develop on Cloud Foundry. You just need to create a dev space, which is a preconfigured development environment with the tools, capabilities, and resources needed for developing your application.

1. Register for an SAP BTP trial account at <https://account.hanatrial.ondemand.com/> and log on afterwards.
2. Set Up SAP Business Application Studio in a trial account.
3. Create a SAP Fiori dev space.

4. Create a Project - either using the wizard or the terminal.

Learn how you can use SAP Business Application Studio together with UI5 Tooling to develop state-of-the-art UI5 apps.

## Tutorials

Depending on the project template you're using you have a project that is more advanced than what you see in the following tutorials. Those tutorials show you the basics and you can start a new and empty project to follow along those SAPUI5 tutorials from the demo kit.

Have a look at the [Get Started: Setup and Tutorials](#) section that contains the following topics and more:

- “Hello World!”
- Data Binding
- Navigation and Routing
- Mock Server
- Worklist App
- Ice Cream Machine

## Essentials

This is reference information that describes the development concepts of SAPUI5 , such as the Model View Controller, data binding, and components.

The following topics are excerpts from the [Essentials](#) section:

- Bootstrapping: Loading and Initializing
- Structuring: Components and Descriptor
- Model View Controller (MVC)
- Data Binding
- Reusing UI Parts: Fragments

## Developing Apps

Create apps with rich user interfaces for modern Web business applications, responsive across browsers and devices, based on HTML5.

The following topics are excerpts from the [Developing Apps](#) section:

- App Templates: Kick Start Your App Development
- App Overview: The Basic Files of Your App
- App Initialization: What Happens When an App Is Started?
- Folder Structure: Where to Put Your Files
- Coding Issues to Avoid
- Securing Apps

## 4.1.4.2 Developing HTML5 Applications in the Cloud Foundry Environment

SAP BTP enables you to access and run HTML5 applications in a cloud environment without the need to maintain your own runtime infrastructure.

HTML5 applications consist of static content that runs on a browser. Then you develop your applications - either in SAP Business Application Studio, or in your own IDE (integrated development environment) - and deploy them to the [HTML5 application repository \[page 392\]](#).

Depending on your backend application setup, you either configure the destinations during development, or define them after deploying the application. Finally, to consume the applications, you can create a site using SAP Cloud Portal service or SAP Launchpad service, build the URL, and define custom domains.

On the Cloud Foundry environment of SAP BTP you can run an application that was uploaded to HTML5 application repository using one of the following options:

- Use the application router that is managed by SAP.  
For more information, see [Managed Application Router \[page 487\]](#).
- Set up and maintain your own application router in your own space.  
For more information, see [Application Router \[page 409\]](#).

Both options allow you to serve static content from HTML5 application repository, authenticate users, rewrite URLs, and forward or proxy requests to other micro services while propagating user information. However, the [managed application router \[page 487\]](#) brings many benefits, such as:

- Simplify and speed up your development and deployment experience
- Save resources by running a serverless HTML5 application, which doesn't require any application runtime
- Lower maintenance efforts by leveraging the most up-to-date routing capabilities
- Meet the changing demand for HTML5 applications by automatically adjusting the service to maintain consistent and predictable performance
- Monitor and run your HTML5 applications from the [HTML5 Applications](#) UI in the SAP BTP cockpit

Therefore, we recommend running your own [application router \[page 409\]](#) only in advanced cases, for example when application router extensibility is required.

### ⓘ Note

When using the [managed application router \[page 487\]](#), you can monitor and run your HTML5 applications from the [HTML5 Applications](#) UI for your subaccount in the SAP BTP cockpit.

For monitoring and running HTML5 applications served by a standalone [application router \[page 409\]](#), you can use the [CF HTML5 applications repository CLI plugin](#) ↗.

## Related Information

[HTML5 Application Repository \[page 392\]](#)

[Application Router \[page 409\]](#)

[Managed Application Router \[page 487\]](#)

## 4.1.4.2.1 HTML5 Application Repository

HTML5 application repository enables central storage of HTML5 applications' static content on the SAP BTP, Cloud Foundry environment. This service can be consumed from the SAP BTP, Cloud Foundry runtime, the SAP BTP, Kyma runtime, Kubernetes, or by creating a service instance on the subaccount level regardless of an environment.

HTML5 applications consist of static content such as HTML, CSS, JavaScript, and other files, that run on a browser. For more information, see [Basic Template](#) and [openui5-basic-template-app](#).

HTML5 application repository allows application developers to manage the lifecycle of their HTML5 applications. In runtime, the repository enables the consuming application, typically the application router, to access HTML5 application static content in a secure and efficient manner. For more information, see [Configure Application Router](#).

## Features

### Zero Down-Time Enablement

- The HTML5 applications are decoupled from the consuming application router. This enables updating the static content of the HTML5 applications without restarting the application router.

### Versioning and Authorization

- Exploration of HTML5 application content by version.
- Access control that is based on private or public authorization.  
When the HTML5 application is public, the service enables sharing this content with consuming application routers from different spaces.

### Availability and Performance

- During runtime, the HTML5 application content is cached and optimized to provide high performance with minimal network load.
- The service provides several instances for a runtime to serve a high load of application requests.

## Environment

This service is available in the following environments:

- Cloud Foundry environment
- Kyma environment

## Multitenancy Support

This service supports multitenancy. It can be used by tenant-aware applications.

## Technical Constraints

- The size of an application deployed to the repository is limited to 100 MB per service instance of the [app-host](#) service plan.
- Since the applications stored in HTML5 application repository can be shared, it is advised not to add personal data to them.
- You can perform up to 50 deployments per minute at design time per SAP BTP subaccount.  
If you exceed this rate limit, the HTTP error code 429 is returned.
- You can make up to 300 requests per second at runtime per SAP BTP subaccount.  
If you exceed this rate limit, the HTTP error code 429 is returned.

## See Also

If you want to learn more about services in the SAP BTP, Cloud Foundry environment, see [Using Services in the Cloud Foundry Environment \[page 248\]](#).

### 4.1.4.2.1.1 Deploying Content

Learn how to deploy your content using the preferred Generic Application Content Deployer or deploy, redeploy, and undeploy content from the HTML5 Application Repository.

#### Note

The HTML5 Application Repository service does not provide the technical capabilities to support the collection, processing, and storage of personal data. The file names used for deploying the HTML5 applications are not intended for personal data.

You use a service instance of the app-host service plan to deploy your applications to the HTML5 Application Repository.

#### Note

If you delete an app-host service instance, the apps that are deployed with this app-host service instance are deleted from the HTML5 Application Repository.

#### Restriction

Delta deployments are not supported.

If you deploy new application content using an existing app-host service instance, the new application content will completely replace the old application content that had been previously deployed with this service instance in the HTML5 Application Repository.

For example, if you have already deployed the applications a, b, and c with the instance 123 and now want to deploy the new applications d and e, you can do one of the following to avoid overwriting the applications a, b, and c:

- Redeploy instance 123 with applications a, b, c, d, and e (with the new and the old applications).
- Create a new instance 456 and use this instance to deploy the applications d and e.

## Related Information

[Deploy Content Using the Generic Application Content Deployer \[page 394\]](#)

[Deploy Content Using HTML5 Application Deployer \[page 396\]](#)

[Redeploy Content \[page 401\]](#)

[Undeploy Content \[page 402\]](#)

### 4.1.4.2.1.1.1 Deploy Content Using the Generic Application Content Deployer

Deploy content from the HTML5 Application Repository using the Generic Application Content Deployer (GACD).

## Prerequisites

- `cf CLI` is installed locally, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).
- The multitarget application (MTA) plug-in for the Cloud Foundry command-line interface to deploy MTAs is installed locally. For more information, see [Install the MultiApps CLI Plugin in the Cloud Foundry Environment \[page 2455\]](#).
- The Cloud MTA Build Tool (MBT) is installed, see [Cloud MTA Build Tool: Download](#) 

## Context

You can deploy your content to the HTML5 Application Repository using the GACD (Generic Application Content Deployer) module. The GACD module has the module type `com.sap.application.content`. This module type enables the deploy plug-in generic application content deploy support. It means that when a module is processed in the cf deploy flow, the deploy service locates the service resource that is required as a target for the deploy and deploys the corresponding `content.zip` file.

## Procedure

1. Create a nested `content.zip` file that contains a zip file for each HTML5 application.

Example of a nested content.zip file

### ↔ Sample Code

```
content.zip
  app1.zip
    manifest.json
    xs-app.json
    ...
  app2.zip
  ...
```

2. Create an MTA development descriptor (`mta.yaml` file).

See [Defining Multitarget Application Development Descriptors \[page 576\]](#).

3. In the MTA development descriptor (`mta.yaml` file), define the deployer module:

### ↔ Sample Code

```
ID: testdeployer
_schema-version: '3.1'
modules:
  - name: ui_deployer
    type: com.sap.application.content
    path: deployer/
    requires:
      - name: uideployer_html5_repo_host
        parameters:
          content-target: true
resources:
  - name: uideployer_html5_repo_host
    parameters:
      service-plan: app-host
      service: html5-apps-repo
    type: org.cloudfoundry.managed-service
version: 0.0.1
```

4. To create an MTA archive (`mtar` file), build your content with the [MTA Build Tool \(MBT\)](#).

Run the following command:

### ↔ Sample Code

```
mbt build
```

For more information on MBT build options, see: [How to build an MTA archive from the project sources](#).

5. Deploy the MTA archive (`mtar` file) using the CLI command `cf deploy`.

Run the following command:

### ↔ Sample Code

```
cf deploy <path-to-mtar-file>
```

For example, from the directory where your mtar file is located, run `cf deploy myapp_0.0.1.mtar`.

## Related Information

[Deploying Content with Generic Application Content Deployment \[page 677\]](#)

### 4.1.4.2.1.1.2 Deploy Content Using HTML5 Application Deployer

Use the HTML5 application deployer module to deploy the content of the HTML5 applications to the HTML5 Application Repository.

## Prerequisites

- `cf CLI` is installed locally, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).
- The multi-target application (MTA) plug-in for the Cloud Foundry command line interface to deploy MTAs is installed locally. For more information, see [Install the MultiApps CLI Plugin in the Cloud Foundry Environment \[page 2455\]](#).

#### Note

It is not recommended to use the HTML5 application deployer to deploy application content that is larger than 10 MB.

For larger application content, please use the [Generic Application Content Deployer \(GACD\) \[page 394\]](#) or configure the [asynchronous content upload \[page 399\]](#).

## Context

The HTML5 application deployer is available on npmjs.com (NPM), see [@sap/html5-app-deployer](#).

You can deploy your content to the HTML5 Application Repository using the HTML5 application deployer npm module.

## Procedure

1. Add the `html5-app-deployer` module as a dependency to your `package.json` file. To do so, navigate to your `package.json` file and execute `npm install` to download the `html5-app-deployer` module from the SAP npm registry.

The basic `package.json` file should look similar to the following example:

### ↔ Sample Code

```
{  
  "name": "myAppDeployer",  
  "engines": {  
    "node": ">=6.0.0"  
  },  
  "dependencies": {  
    "@sap/html5-app-deployer": "2.0.1"  
  },  
  "scripts": {  
    "start": "node node_modules/@sap/html5-app-deployer/index.js"  
  }  
}
```

The `start` script is mandatory as it is executed after the deployment of the application.

2. In the `html5-app-deployer` structure, create a `resources` folder and add the static content that you want to deploy. In the `resources` folder, add one folder for each application you want to deploy. For each application you want to deploy, provide a `manifest.json` and `xs-app.json` file at root level.

If you want to deploy more than one application to the same app host instance, you can add multiple zip archives to the resources folder.

### ↔ Sample Code

```
myAppsDeployer  
+ node_modules  
- resources  
- app1  
  index.html  
  manifest.json  
  xs-app.json  

```

- a. The `manifest.json` file contains `sap.app.id` and `sap.app.applicationVersion.version`, which are used in the HTML5 Application Repository as `applicationName` and `applicationVersion`.

### ⓘ Note

The format of the application version is `xx.xx.xx`.

### ↔ Sample Code

```
manifest.json
```

```
{
  "_version": "1.7.0",
  "sap.app": {
    "id": "appl",
    "type": "application",
    "i18n": "i18n/i18n.properties",
    "applicationVersion": {
      "version": "1.0.0"
    }
  }
}
```

- b. The `xs-app.json` file is used to support application routing.

#### ↔ Sample Code

```
xs-app.json
{
  "welcomeFile": "index.html",
  "authenticationMethod": "route",
  "routes": [
    {
      "source": "^/be$",
      "destination": "simpleui_be",
      "authenticationType": "xsuaa"
    },
    {
      "source": "^/ui(/.*)",
      "target": "$1",
      "service": "html5-apps-repo-rt",
      "authenticationType": "xsuaa"
    }
  ]
}
```

3. Create an `mta.yaml` (MTA development descriptor) file:

- Under `resources`, add a resource definition to create an `html5-apps-repo` service instance of the `app-host` service plan.
- Under `modules`, add your app.
- Add a dependency to the `html5-apps-repo` service and the `app-host` service plan instance using the `required` statement.

#### ↔ Sample Code

```
ID: html5.repo.deployer.myHTML5App
_schema-version: '2.0'
version: 0.0.3
modules:
- name: myHTML5App_app-deployer
  type: com.sap.html5.application-content
  path: deployer/
  requires:
    - name: myHTML5App_app-host

resources:
- name: myHTML5App_app-host          //Resource name
  type: org.cloudfoundry.managed-service
  parameters:
    service: html5-apps-repo          //Service name
    service-plan: app-host            //Service plan
    service-name: myHTML5App_app-host //Service instance name
```

For a general description of MTA descriptors, see [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#).

4. Build and deploy the mta.yaml (MTA development descriptor).

For more information about the build and deployment of MTA descriptors, see [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#).

The deployment starts the following:

- Create an HTML5 Application Repository service instance of the app-host service plan.
- Create an HTML5 application deployer application, which uses the HTML5 application deployer npm module.
- Bind the app-host service instance to the HTML5 application deployer application.
- Start the HTML5 application deployer application:
  - Create a zip archive for each application in resources folder.
  - Create a client credential token from the app-host service instance credentials.
  - Deploy the content to the HTML5 application repository: passing on the zip archives and the client credential token.
- Stop the HTML5 application deployer application.

#### **4.1.4.2.1.1.2.1 Asynchronous Upload Using the HTML5 Application Deployer**

You can specify that the upload of content is performed asynchronously by adding the environment variable ASYNC\_UPLOAD to the manifest.yaml or mta.yaml files.

The asynchronous upload is especially useful, if you want to trigger the upload of service instances with a large content (more than 10 MB). A synchronous upload of service instances with a large content can result in health check errors or a connection timeout during the upload. You can avoid this by using the asynchronous upload.

During an asynchronous upload, the html5 applications content is handled synchronously to the HTML5 Application Repository, while the internal file validation and processing is performed asynchronously. To verify that the upload has completed successfully, you have to check the HTML5 Application Deployer logs.

#### **4.1.4.2.1.1.2.2 Automatic Creation of Destination Configurations**

If you use the HTML5 application deployer together with an application router managed by SAP for the Kyma runtime, you can enable that the required destination configurations pointing to the service instances are created automatically.

To enable the automatic creation of destination configurations, add the environment variable SAP\_CLOUD\_SERVICE in the deployment.yaml or Helm chart of the Kyma project and provide the value of the sap.cloud.service property from the manifest.json file of the HTML5 application that you want to deploy. The destination configurations can point to instances of the following services:

- An HTML5 Application Repository service instance of the app-host service plan (mandatory)
- An SAP Authorization and Trust Management (XSUAA) service instance (optional)  
To create a destination that points to an SAP Authorization and Trust Management (XSUAA) service instance, the SAP Authorization and Trust Management (XSUAA) service instance and a destination service instance must be bound to the HTML5 application deployer.
- A business service instance (optional)
- One or more backend destinations that point to cloud or on-premise backend applications.  
In addition to the SAP\_CLOUD\_SERVICE environment variable, also add the environment variable BACKEND\_DESTINATIONS to create backend destinations.

### Note

To enable app-to-app navigation, you also have to add the environment variable IAS\_DEPENDENCY\_NAME and provide the name of the dependency that has been configured for the Identity Authentication token exchange that is required for app-to-app navigation. For more information about how to configure the dependency for app-to-app navigation, see [Consume an API from Another Application](#).

You would typically use the automatic creation of destination configurations in Kubernetes if the HTML5 application deployer application has been previously uploaded as a docker image to Artifactory or Docker Hub. See, for example, this Kubernetes deployment:

### Sample Code

```
--  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: html5appdeployer  
  namespace: default  
  labels:  
    app: html5appdeployer  
spec:  
  replicas: 1  
  selector:  
    matchLabels:  
      app: html5appdeployer  
  template:  
    metadata:  
      labels:  
        app: html5appdeployer  
    spec:  
      containers:  
        - image: html5-apps-repo.docker.repositories.sap.ondemand.com/myapp-  
          html5-app-deployer:1.0  
          name: html5appdeployer  
          volumeMounts:  
            - name: html5-repo-app-host-volume  
              mountPath: "/etc/secrets/sapcp/html5-apps-repo/myapp-app-host-  
instance"  
              readOnly: true  
            - name: xsuaa-volume  
              mountPath: "/etc/secrets/sapcp/xsuaa/myapp-xsuaa-instance"  
              readOnly: true  
            - name: destination-volume  
              mountPath: "/etc/secrets/sapcp/destination/myapp-destination-  
instance"  
              readOnly: true  
      env:  
        - name: PORT  
          value: "5000"
```

```

      - name: SAP_CLOUD_SERVICE
        value: "com.sap.test.service"
      - name: BACKEND_DESTINATIONS
        value: "[{
          \"Name\": \"myapp-backend\",
          \"Description\": \"My application backend\",
          \"Type\": \"HTTP\",
          \"ProxyType\": \"Internet\",
          \"URL\": \"https://<backendApplicationHost>/\",
          \"Authentication\": \"NoAuthentication\",
          \"HTML5ForwardAuthToken\": true}]"
    imagePullSecrets:
      - name: backend-dockersecret
    volumes:
      - name: html5-repo-app-host-volume
        secret:
          secretName: myapp-app-host-binding
      - name: xsuaa-volume
        secret:
          secretName: myapp-xsuaa-binding
      - name: destination-volume
        secret:
          secretName: myapp-destination-binding

```

#### 4.1.4.2.1.1.3 Redeploy Content

You can redeploy changed content to the existing app-host service instance.

After making changes to the static content files of HTML5 applications, you can redeploy the new content to the already existing app-host service instance of the HTML5 application repository. All content referenced by the app-host service instance ID is replaced by the new content. If you want to keep more than one version of your HTML5 application on the repository, you deploy all versions, the new versions and the old versions that you want to keep, to the same app-host service instance ID.

#### Example: Deploying two Versions to the Repository

- Application **app1** with version 1.0.0 has been deployed to the repository. This app matches a back-end application with version 3.0.0
- For the app, a new back-end version 4.0.0 is available. Therefore, the application with version 2.0.0 is developed that matches the new back-end version.
- As customers may still have a backend with version 3.0.0, **app1** with version 1.0.0 cannot be dropped. On the repository, both versions shall be available so the customer can decide which one to use depending on their back-end version.

##### ↔ Sample Code

```

myAppsDeployer
  + node_modules
  - resources
    - app1
      index.html
      manifest.json

```

```
  xs-app.json
  - app2
    ...
  package.json
  manifest.yaml
```

#### 4.1.4.2.1.1.4 Undeploy Content

To undeploy content you need to delete the content from the repository and delete the `app-host` service plan instance.

##### Procedure

1. Open the Cloud Foundry command line interface (CLI).
2. Undeploy the `*.mtar` file using the CLI command: `cf undeploy` and add the `--delete-services` `--delete-service-keys` option.

Example: `cf undeploy html5.repo.deployer.myHTML5App --delete-services --delete-service-keys`

- Use the `-delete-services` option, to delete the `app-host` service plan instance and the application content from HTML5 application repository.
- For the `cf undeploy` command, you need the `mta id`. To get the `mta id`, do one of the following:
  - Call `cf mtas`.
  - Check the `mtad.yaml ID`.

##### ⓘ Note

If you do not want to use the `-delete-services` option, you can delete the `app-host` service plan instance manually using the CLI command: `cf delete-service SERVICE-NAME`.

#### 4.1.4.2.1.2 Service Plans

The HTML5 Application Repository service comprises the following service plans:

- **`app-host`**  
Use this service plan to deploy HTML5 applications to the repository. For more information, see [Deploying Content \[page 393\]](#).
- **`app-runtime`**  
Use this service plan to consume HTML5 applications stored in the repository. For more information, see [Consuming Content \[page 403\]](#).

### 4.1.4.2.1.3 Consuming Content

Use the `app-runtime` service plan to consume HTML5 applications from the HTML5 Application Repository.

#### Context

Application router can consume content from the HTML5 Application Repository by binding an HTML5 Application Repository service instance with `app-runtime` plan to the application router. In addition, the routing to the HTML5 Application Repository service is configured in the `xs-app.json` file.

#### Procedure

1. Create a service instance of `app-runtime` service plan.

Example: `cf create-service html5-apps-repo app-runtime MyHtml5AppsRepoRuntimeInstance`

2. Bind the HTML5 Application Repository runtime instance to the application router in the `manifest.yaml` file.

##### ↔ Sample Code

```
applications:  
- name: myCRMApp  
  memory: 256M  
  services:  
    - MyHtml5AppsRepoRuntimeInstance
```

3. Push the application with the following command: `cf push`
4. To redirect requests to the HTML5 Application Repository runtime, configure the routing to the HTML5 Application Repository service in your `xs-app.json` file.

##### ↔ Sample Code

```
{  
  
  "source": "^(/.* )",  
  
  "target": "$1",  
  
  "service": "html5-apps-repo-rt",  
  
  "authenticationType": "xsuaa"  
}
```

5. HTML5 application content can be consumed from the application router using an URL in the following format: `https://<host>. <domain>/<appName>-<appVersion>/<resourcePath>`

#### ⓘ Note

`appName` - the `manifest.json app.id` **without the dots**

`appVersion` - the `manifest.json applicationVersion.version`. The `appVersion` is optional. If not provided, the latest version will be used.

### 4.1.4.2.1.4 Security and Data Privacy Configuration

This section provides information about security and data privacy.

- Identity Provider and Identity Management

You can use the User Account and Authentication (UAA) server for user authentication. In Cloud Foundry, a service is created for this configuration. By using the standard service binding mechanism, the content of this configuration is available in the `VCAP_SERVICES` environment variable.

As an alternative to the User Account and Authentication (UAA) service , you can also use the Identity Authentication service of the SAP Cloud Identity Services for authentication.

- Authorization Configuration

To obtain the access token to authorize API requests, the client credentials authorization flow is used.

- Data Protection and Data Privacy

The HTML5 application repository service applies the legally required data protection and privacy measures to data that is stored on behalf of customers. These measures include, for example, the need to know principle during support processes, or data isolation between spaces.

The HTML5 application repository runtime is a microservice responsible for providing read access to HTML5 applications during runtime. When the HTML5 application is public, the service enables sharing this content with consuming application routers from different spaces.

#### ⓘ Note

The HTML5 application repository service does not provide the technical capabilities to support the collection, processing, and storage of personal data. The file names used for deploying the HTML5 applications are not intended for personal data.

### 4.1.4.2.1.4.1 Configure the Credential Type for HTML5 Application Repository Service Instances

When you create an HTML5 application repository service instance key or bind a service instance to an application, values for `client_id` and `client_secret` are generated. The `client_secret` is the equivalent to a password.

As of January 28, 2021, the credential type `binding_secret` is automatically used for the `client_secret` value of new HTML5 application repository service instance keys. This increases the security because the credential type `binding_secret` creates a new `client_secret` for each binding and service key.

If you have older HTML5 application repository service instances, the `client_secret` value is the same for all bindings and service keys because the HTML5 application repository still uses the credentials type

`instance_secret` for older HTML5 application repository service instances. To configure the more secure credential type `binding_secret` for your older HTML5 application repository service instances, please follow the instructions in [Configure the Credential Type binding\\_secret for HTML5 Application Repository Service Instances \[page 405\]](#).

As an alternative to the credential type `binding_secret`, you can also use the credential type `x.509` to authenticate your apps with X.509 certificates. For more information, see [Credential Type x509 for HTML5 Application Repository Service Instances \[page 406\]](#).

## Configure the Credential Type `binding_secret` for HTML5 Application Repository Service Instances

1. Extract the UUID part of the `xsappname` from the of the existing instance. For example:

1. **Sample Code**

```
cf create-service-key <serviceInstanceName> <serviceInstanceKeyName>
```

2. **Sample Code**

```
cf service-key <serviceInstanceName> <serviceInstanceKeyName>
.....
"clientsecret": "HksDYy9RxYFUQeOHIZprqBam3vQ=", //Old instance secret
...
"xsappname": "fec39a25-a570-46b0-8ac9-1691f8d663cc!b6711|html5-apps-
repo-uaa!b6711",
.....
```

3. Copy the UUID. In this example: **fec39a25-a570-46b0-8ac9-1691f8d663cc**

2. Create a file `instance-xs-security.json`, such as the following:

- Sample Code**

```
{
  "xs-security": {
    "xsappname": "fec39a25-a570-46b0-8ac9-1691f8d663cc",
    "oauth2-configuration": {
      "credential-types": ["binding-secret"]
    }
  }
}
```

3. Update the service instance:

- Sample Code**

```
cf update-service <serviceInstanceName> -c instance-xs-security.json
```

4. Check that the instance now creates binding secrets as follows:

- Sample Code**

```
cf create-service-key <serviceInstanceName>
<serviceInstanceKeyNameNew>
```

## 2. ⚡ Sample Code

```
cf service-key <serviceInstanceName> <serviceInstanceKeyNameNew>
.....
"clientid": "sb-fec39a25-a570-46b0-8ac9-1691f8d663cc!b6711|html5-apps-
repo-uaa!b6711",
"clientsecret":
"8c902673-00d4-471f-9ebb-4463d4073fda$7NIo5batSFZCF1RNGcpNkRfo4mDj6vq-
CElDFIcvwg=", //New binding secret (long)
"identityzone": ....,
"identityzoneid": ....,
.....
```

## Credential Type x509 for HTML5 Application Repository Service Instances

To use the HTML5 application repository service and authenticate with X.509 certificate, you need to acquire the required key, certificate, and base URLs. These are provided when you create an instance of the HTML5 application repository service and a binding or a service key for the instance.

When you bind an HTML5 applications repository service instance to an application, you have to explicitly provide the required configuration parameters for the authentication with x.509:

```
cf bind-service <application> <service-instance> -c parameters.json
```

The same applies if you create service keys:

```
cf create-service-key <service-instance> <key-name> -c parameters.json
```

The configuration for the credential type x509 must be wrapped with xsuaa. You have to use exactly the following format for the configuration:

```
{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "key-length": 2048, // specifies the byte length of the generated private
key, // defaults to 2048
      "validity": 7, // specifies the number of time units in validity-type,
// defaults to 7, thus the complete validity defaults to 7
days
      "validity-type": "DAYS" // specifies the validity time unit,
// only DAYS, MONTHS and YEARS are supported,
// defaults to DAYS
    }
  }
}
```

### ⓘ Note

The following applies to existing HTML5 application repository service instances:

Before you create a binding to a service instance or a service key as described above, you have to update of the service instance of your existing HTML5 application repository service instance. Use parameters to provide the configuration during the update of the existing service instance:

```
cf update-service <service-instance> -c update-instance-xs-security.json
```

To configure the credential type x509 for existing HTML5 application repository service Instances, please follow the instructions below:

## Configuring Credential Type x509 for Existing HTML5 Application Repository Service Instances

To configure the credential type x509 for your existing HTML5 application repository service instances, perform the following steps:

1. Extract the UUID part of the xsappname from the service key or the binding for the service instance. For example:

### Sample Code

```
cf service-key <serviceInstanceName> <existingServiceInstanceKeyName>
.....
"clientsecret": "HksDYy9RxFUQeOHIZprqBam3vQ=", //Old instance secret
...
"xsappname": "fec39a25-a570-46b0-8ac9-1691f8d663cc!b6711|html5-apps-
repo-uaa!b6711",
.....
```

2. Copy the UUID. In this example: **fec39a25-a570-46b0-8ac9-1691f8d663cc**
2. Create a file update-instance-xs-security.json with the following format:

### Sample Code

```
{
  "xs-security": {
    "xsappname": "fec39a25-a570-46b0-8ac9-1691f8d663cc", //The UUID that
    was extracted before
    "oauth2-configuration": {
      "credential-types": [ "x509" ]
    }
  }
}
```

3. Update the service instance:  
`cf update-service <serviceInstanceName> -c update-instance-xs-security.json`
4. Create a file parameters.json, such as the following:

### Sample Code

```
{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "key-length": 2048,
      "validity": 7,
      "validity-type": "DAYS"
    }
  }
}
```

(Note that the values of the "x509" properties used here are just examples.)

5. Create a binding or a service key with the credential type x509 configuration (parameters.json file). For example:

```
cf create-service-key <serviceInstanceName> <serviceInstanceKeyNameNew> -c
parameters.json
```

6. Check that the instance now creates a certificate and that the credential-type is "x509", as follows:

```
cf service-key <serviceInstanceName> <serviceInstanceKeyNameNew>
The following binding is generated as a result:
```

```
{
  .....
  "certificate": "-----BEGIN CERTIFICATE-----\n-----END CERTIFICATE-----\n-----BEGIN CERTIFICATE-----\n-----END CERTIFICATE-----\n-----BEGIN CERTIFICATE-----\n-----END CERTIFICATE-----\n",
  "clientid": ".....",
  "credential-type": "x509",
  .....
}
```

## 4.1.4.2.1.4.2 Auditing and Logging Information

Here you can find a list of the security events that are logged by the HTML5 application repository.

Events written in audit logs

Event grouping	What events are logged	How to identify related log events	Additional information
Authentication failure	Authentication failed because of an invalid token	Message: <i>Authentication failure status: 401</i>	
	Authentication failed because the token has expired	Message: <i>Authentication failure status: 403</i>	
Attempt to access an application with an invalid app-host ID	Attempt to access an application with an invalid app-host ID	Message: <i>Invalid App Host ID</i>  If there was an attempt to get an application that was deployed by a developer in another space.	
Attempt to access a private application from another space	Attempt to access a private application from another space	Message: <i>Attempt to access private application. application name : ' + appKey + ', app-host : ' + dtServiceInstanceId</i>	
Deletion of an application (Deletion of an app-host service instance from the HTML5 application repository)	Deletion of data started	Message: <i>Deletion of data started for SPACE &lt;space&gt; in ORG &lt;org&gt;</i>	
	Deletion of data completed	Message: <i>Deletion of data completed for SPACE &lt;space&gt; in ORG &lt;org&gt;</i>	

Event grouping	What events are logged	How to identify related log events	Additional information
	Deletion of data failed	Message: <i>Deletion of data failed for SPACE &lt;space&gt; in ORG &lt;org&gt;</i>	

The following information is described in the table columns:

- *Event grouping* - Events that are logged with a similar format or are related to the same entities.
- *What events are logged* - Description of the security or data protection and privacy related event that is logged.
- *How to identify related log events* - Search criteria or key words, that are specific for a log event that is created along with the logged event.
- *Additional information* - Any related information that can be helpful.

## Related Information

[Audit Logging in the Cloud Foundry Environment](#)

[Audit Logging in the Neo Environment](#)

### 4.1.4.2.2 Application Router

The application router is the single point-of-entry for an application running in the Cloud Foundry environment on SAP BTP. The application router is used to serve static content, authenticate users, rewrite URLs, and forward or proxy requests to other micro services while propagating user information.

You can set up and run your own application router or you can use the application router that is managed by SAP (for more information see [Managed Application Router \[page 487\]](#)).

We recommend running your own application router only in advanced cases, for example when application router extensibility is required. The application router is available on the following sites:

- As a library on npmjs.com (NPM), see [@sap/approuter](#)
- As a container image on Docker Hub, see <https://hub.docker.com/r/sapse/approuter>

## Related Information

[Developing HTML5 Applications in the Cloud Foundry Environment \[page 391\]](#)

[Setting Up Your Own Application Router \[page 410\]](#)

[Resource Files \[page 413\]](#)

[Application Router Configuration \[page 414\]](#)

[Routing Configuration File \[page 420\]](#)

[Application Routes and Destinations \[page 450\]](#)

[Environment Variables \[page 463\]](#)

[Multitenancy \[page 480\]](#)

[Extending the Application Router \[page 482\]](#)

[Extension API of the Application Router \[page 485\]](#)

## 4.1.4.2.2.1 Setting Up Your Own Application Router

This section describes how you can set up your own application router.

### Prerequisites

- cf CLI is installed locally, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).
- Node.js is installed and configured locally, see [npm documentation](#).
- The SAP npm registry, which contains the Node.js package for the application router, is configured:  
`npm config set @sap:registry https://registry.npmjs.org`

### Context

The application router is configured in the `xs-app.json` file. The `package.json` file contains the start command for the application router, a list of the package dependencies, and the supported Node.js versions.

### Procedure

1. Create the application resource-file structure.

For example, `/path/<myAppName>/`

2. Create a subfolder for the static Web resources module.

The subfolder for the Web-resources module must be located in the application root folder, for example, `/path/<myAppName>/web`.

#### → Tip

The Web-resource module uses `@sap/approuter` as a dependency; the Web-resources module also contains the configuration and static resources for the application.

3. Create the subfolder for the application's static resources.

Static resources can, for example, include the following file and components: `index.html`, style sheets (`.css` files), images and icons. Typically, static resources for a Web application are placed in a subfolder of the Web module, for example, `/path/<myAppName>/web/resources`.

4. Create the application-router configuration file.

The application router configuration file `xs-app.json` must be located in the application's Web-resources folder, for example, `/path/<MyAppName>/web`.

#### ↔ Sample Code

```
<myAppName>
| - web/
|   | - xs-app.json           # Application descriptors
|   | - package.json          # Application routes configuration
|   \- resources/             # Application router details/dependencies
```

5. Define details of the application's routes, destinations, and security scopes.

The contents of the `xs-app.json` file must use the required JSON syntax. For more information, see [Routing Configuration File \[page 420\]](#).

a. Create the required destinations configuration.

#### ↔ Sample Code

```
/path/<myAppName>/web/xs-app.json.

{
  "welcomeFile": "index.html",
  "routes": [
    {
      "source": "/sap/ui5/1(.*)",
      "target": "$1",
      "localDir": "sapui5"
    },
    {
      "source": "/rest/addressbook/testdataDestructor",
      "destination": "backend",
      "scope": "node-hello-world.Delete"
    },
    {
      "source": "/rest/.*",
      "destination": "backend"
    },
    {
      "source": "^/(.*)",
      "localDir": "resources"
    }
  ]
}
```

b. Add the routes (destinations) for the specific application (for example, `node-hello-world`) to the `env` section of the application's deployment manifest (`manifest.yml`).

Every route configuration that forwards requests to a micro service has property `destination`. The `destination` is a name that refers to the same name in the destinations configuration. The destinations configuration is specified in an environment variable passed to the approuter application.

#### ↔ Sample Code

```
<myAppName>/manifest.yml

- name: node-hello-world
  routes:
    - route: myHost-node-hello-world.xsapps.acme.ondemand.com
```

```

memory: 100M
path: web
env:
  destinations: >
    [
      {
        "name": "backend",
        "url": "http://myHost-node-hello-world-
backend.xsapps.acme.ondemand.com",
        "forwardAuthToken": true
      }
    ]

```

- Add a package descriptor (`package.json`) for the application router to the root folder of your application's Web resources module (`web/`) and execute `npm install` to download the `approuter` npm module from the SAP npm registry.

The package descriptor describes the prerequisites and dependencies that apply to the application router and starts the application router, too.

#### ↔ Sample Code

```

<myAppName>
| - web/                               # Application descriptors
|   | - xs-app.json                    # Application routes configuration
|   | - package.json                  # Application router details/dependencies
|   \- resources/

```

The basic `package.json` file for your Web-resources module (`web/`) should look similar to the following example:

#### ↔ Sample Code

```

{
  "name": "node-hello-world-approuter",
  "dependencies": {
    "@sap/approuter": "5.1.0"
  },
  "scripts": {
    "start": "node node_modules/@sap/approuter/approuter.js"
  }
}

```

#### → Tip

The start script (for example, `approuter.js`) is mandatory; the start script is executed after application deployment.

## Related Information

[Managed Application Router \[page 487\]](#)

## 4.1.4.2.2 Resource Files

The routing configuration for an application is defined in one or more destinations.

The application router is used to serve static content, authenticate users, rewrite URLs, and forward or proxy requests to other micro services while propagating user information. The following table lists the resource files used to define routes for multi-target applications:

Application-Router Resource Files Overview

File	Description	Mandatory
package.json	The package descriptor is used by the Node.js package manager ( <code>npm</code> ) to start the application router; in the <code>"dependencies": {}</code> section	Yes
xs-app.json	The application descriptor contains the configuration used by the application router (for example, destinations for request forwarding)	Yes
resources/	A folder that contains all static resources which should be served by the application router. If no <code>resources/</code> folder is present, the application router will not serve any static content. However, it still forwards requests to the configured destinations.	No
<p>→ Tip</p> <p>If you have a static resources folder name in the <code>xs-app.json</code> file, we recommend that you use <code>localDir</code> as default.</p>		
local-destinations.json	Provides the required destinations information for local development	No
default-services.json	Defines the configuration for one or more special User Account and Authentication (UAA) services for local development	-

### ↔ Sample Code

`xs-app.json`

```
{  
  "source": "^/web-pages/(.*)$"  
  "localDir": "my-static-resources"  
}
```

### 4.1.4.2.2.3 Application Router Configuration

A file that contains the configuration information used by the application router.

The application router configuration file is named `xs-app.json` and its content is formatted according to JavaScript Object Notation (JSON) rules.

When a business application consists of several different apps (microservices), the application router is used to provide a single entry point to that business application. The application router is responsible for the following tasks:

- Dispatch requests to back-end microservices (reverse proxy)
- Authenticate users
- Serve static content
- In multitenancy scenarios, derive the tenant information from the URL and forward the information to the XS UAA service so that the authentication request is redirected to the appropriate tenant-specific Identity Provider (IdP), for example, using SAML “Bearer Assertions”.

#### → Tip

The different applications (microservices) are the destinations to which the incoming requests are forwarded. The rules that determine which request should be forwarded to which destination are called routes. For every destination there can be more than one route.

## User Authentication Services

For the user authentication, you can use User Account and Authentication (UAA) service or the Identity Authentication service.

### User Account and Authentication (UAA)

User authentication is performed by the User Account and Authentication (UAA) server. In the run-time environment (on-premise and in the Cloud Foundry environment), a service is created for the UAA configuration; by using the standard service-binding mechanism, the content of this configuration is available in the `<VCAP_SERVICES>` environment variable, which the application router can access to read the configuration details.

#### ⓘ Note

The UAA service should have `xsuaa` in its tags or the environment variable `<UAA_SERVICE_NAME>` should be defined, for example, to specify the **exact** name of the UAA service to use.

A calling component accesses a target service by means of the application router only if there is no JSON Web Token (JWT) available, for example, if a user invokes the application from a Web browser. If a JWT token is already available, for example, because the user has already been authenticated, or the calling component uses a JWT token for its own OAuth client, the calling component calls the target service directly; it does not need to use the application router.

### Note

The application router does not “hide” the back-end microservices in any way; they remain directly accessible when bypassing the application router. So the back-end microservices must protect all their end points by validating the JWT token and implementing proper authorization scope checks.

The application router supports the use of the `$XSAPPNAME` placeholder, which you can use in your route configuration, for example, in the `scope` property for the specified route. The value of `$XSAPPNAME` is taken from the UAA configuration (for example, the `xsappname` property). For more information, see [Routing Configuration File \[page 420\]](#).

### Identity Authentication Service

As an alternative to the User Account and Authentication service (UAA), you can also use Identity Authentication. This service provides authentication and single sign-on for users in the cloud. For more information, see the product page for [SAP Cloud Identity Services - Identity Authentication](#) on the SAP Help Portal.

## Headers

If back end nodes respond to client requests with URLs, these URLs need to be accessible for the client. For this reason, the application router passes the following `x-forwarding-*` headers to the client:

- `x-forwarded-host`  
Contains the **host** header that was sent by the client to the application router
- `x-forwarded-proto`  
Contains the **protocol** that was used by the client to connect to the application router
- `x-forwarded-for`  
Contains the **address** of the client which connects to the application router
- `x-forwarded-path`  
Contains the original **path** which was requested by the client from the approuter

### Caution

If the application router forwards a request to a destination, it blocks the header `host`.

“Hop-by-hop” headers are meaningful only for a single transport-level connection; these headers are not forwarded by the application router. The following headers are classified as “Hop-By-Hop” headers:

- Connection
- Keep-Alive
- Public
- Proxy-Authenticate
- Transfer-Encoding
- Upgrade

You can configure the application router to send additional HTTP headers, for example, either by setting it in the `httpHeaders` environment variable or in a `local-http.headers.json` file.

## ↔ Sample Code

```
local-http.headers.json  
[  
  {  
    "X-Frame-Options": "ALLOW-FROM http://localhost"  
  },  
  {  
    "Test-Additional-Header": "1"  
  }  
]
```

### ⚠ Caution

For security reasons, the following headers must not be configured:

- Authorization
- Set-Cookie
- Cookie

## Sessions

The application router establishes a session with the client (browser) using a session cookie. The application router intercepts all session cookies sent by back-end services and stores them in its own session. To prevent collisions between the various session cookies, back-end session cookies are not sent to the client. On request, the application router sends the cookies back to the respective back-end services so the services can establish their own sessions.

### ⓘ Note

Non-session cookies from back-end services **are** forwarded to the client, which might cause collisions between cookies. Applications should be able to handle cookie collisions.

### ⚠ Restriction

You must not use multiple destinations with the same URL in an application router user session!

The application router stores the session cookies from the backend services in a user session according to the destination URLs for the backend services. If there are multiple destinations with the same URL, the application router cannot correctly send the cookies back to the backend services.

To avoid conflicts, the destination URLs must have different domains.

If there is no session in the application router, either because there has been a session timeout or because no session has been created yet, and if the incoming request matches a non-public route, the application router triggers a redirect to the authentication service (UAA or IAS).

After a successful login, a redirect back to application router takes place using the login callback endpoint, which triggers the creation of a new session. If the incoming request is an AJAX request (has the request header `X-Requested-With: XMLHttpRequest`) or if the HTTP verb is not `GET` and no session exists (there

is no session cookie and the request doesn't have an x-approuter-authorization header), the application router returns the response code 401 – Unauthorized. This enables the client application to handle the 401 response before it navigates to the authentication service. For example, the application can store data entered by the user and prevent data loss. When the handling of the 401 response is completed, the client application should send a request without an `xmlhttprequest` object to trigger the application router authentication flow.

## Session Contents

A session established by the application router typically contains the following elements:

- Redirect location  
The location to redirect to after logon; if the request is redirected to a UAA logon form, the original request URL is stored in the session so that, after successful authentication, the user is redirected back to it.
- CSRF token  
The CSRF token value if it was requested by the clients. For more information about protection against Cross Site Request Forgery see [CSRF Protection \[page 418\]](#) below.
- OAuth token  
The JSON Web Token (JWT) fetched from the User Account and Authentication service (UAA) and forwarded to back-end services in the `Authorization` header. The client never receives this token. The application router refreshes the JWT automatically before it expires (if the session is still valid). By default, this routine is triggered 5 minutes before the expiration of the JWT, but it can also be configured with the `<JWT_REFRESH>` environment variable (the value is set in minutes). If `<JWT_REFRESH>` is set to 0, the refresh action is disabled.
- OAuth scopes  
The scopes owned by the current user, which are used to check if the user has the authorizations required for each request.
- Back-end session cookies  
All session cookies sent by back-end services.

## Scaling

The application router keeps all established sessions in local memory, and if multiple instances of the application router are running, there is no synchronization between the sessions. To scale the application router for multiple instances, session stickiness is used so that each HTTP session is handled by the same application router instance.

## Sizing and Memory Configuration

The application-router process should run with at least 256MB memory. The amount of memory actually required depends on the application the router is serving. The following aspects have an influence on the application's memory usage:

- Concurrent connections
- Active sessions

- Size of the Java Web Token
- Back-end session cookies

You can use the start-up parameter `max-old-space-size` to restrict the amount of memory used by the JavaScript heap. The default value for `max-old-space-size` is less than 2GB. To enable the application to use all available resources, the value of `max-old-space-size` should be set to a number equal to the memory limit for the whole application. For example, if the application memory is limited to 2GB, set the heap limit as follows, in the application's `package.json` file:

#### ↔ Sample Code

```
"scripts": {
  "start": "node --max-old-space-size=2048 node_modules/@sap/approuter/
approuter.js"
}
```

If the application router is running in an environment with limited memory, set the heap limit to about 75% of available memory. For example, if the application router memory is limited to 256MB, add the following command to your `package.json`:

#### ↔ Sample Code

```
"scripts": {
  "start": "node --max-old-space-size=192 node_modules/@sap/approuter/
approuter.js"
}
```

#### ⓘ Note

For detailed information about memory consumption in different scenarios, see the Sizing Guide for the Application Router located in `approuter/approuter.js/doc/sizingGuide.md`.

## CSRF Protection

The application router enables CSRF protection for any HTTP method that is not `GET` or `HEAD` and the route is not public. A path is considered public, if it does not require authentication. This is the case for routes with `authenticationType: none` or if authentication is disabled completely via the top level property `authenticationMethod: none`.

To obtain a CSRF token one must send a `GET` or `HEAD` request with a `x-csrf-token: fetch` header to the application router. The application router will return the created token in a `x-csrf-token: <token>` header, where `<token>` will be the value of the CSRF token.

If a CSRF protected route is requested with any of the above mentioned methods, `x-csrf-token: <token>` header should be present in the request with the previously obtained token. This request must use the same session as the fetch token request. If the `x-csrf-token` header is not present or is invalid, the application router will return status code “403 - Forbidden”.

## Cloud Connectivity

The application router supports integration with SAP Connectivity service. The connectivity service enables you to manage proxy access to Cloud Connector, which you can use to create tunnels for connections to systems located in private networks, for example, on-premise. To use the connectivity feature, you must create an instance of the connectivity service and bind it to the `Approuter` application.

In addition, the relevant destination configurations in the `<destinations>` environment variable must have the proxy type "onPremise", for example, `"proxyType": "onPremise"`. You must also ensure that you obtain a valid XSUAA logon token for the XS advanced User Account and Authentication service.

## Troubleshooting

The application router uses the `cf-nodejs-logging-support` package, giving you access to all its logging control features. For example, you can set all logging and tracing to the finest level by setting the `CF_NODEJS_LOGGING_LEVEL` environment variable to `debug`.

If you've deployed the application on Cloud Foundry, you can change the log level and restart the application using the following command:

### Sample Code

```
cf set-env <application-name> CF_NODEJS_LOGGING_LEVEL debug
```

You can enable additional traces of incoming and outgoing requests by setting the `<REQUEST_TRACE>` environment variable to `true`. When enabled, basic information will be logged for every incoming and outgoing request of the application router. Note that this could impact performance.

Some libraries used by the `cf-nodejs-logging-support` package use other tracing mechanisms. For example, you can use the `debug` package. By setting the `'DEBUG'` environment variable, you can enable additional traces.

### Note

Enabling the `debug` log level could lead to a very large amount of data being written to the application logs and trace files. The asterisk wild card (\*) enables options that trace sensitive data that is then written to the logs and traces.

In addition, you can enable internal Node.js traces via the `'NODE_DEBUG'` environment variable.

### Note

Be cautious when enabling some of these options as they may trace security-sensitive data.

The `cf-nodejs-logging-support` package sets the `x-request-id` header in the application router's responses. This is useful if you want to search entries related to a particular request execution in the logs and traces of the application router. Note that the application router doesn't change the headers it receives from the backend and forwards to the client. If the backend is a Node.js application using the `cf-nodejs-logging-support` package (and also sets the `x-request-id` header), the header value that the client receives will be the one from the backend, not the one from the application router.

## ⓘ Note

You can also use the `XS\_APP\_LOG\_LEVEL` environment variable for backward compatibility in SAP HANA extended application services, advanced model (XS advanced). If you've deployed the application on the XS advanced runtime for SAP HANA Platform 2.0, you can change the log level and restart the application using the following command:

### ↪ Sample Code

```
xs set-env <application-name> XS_APP_LOG_LEVEL debug
```

## → Tip

Logging levels are application-specific and case-sensitive; they can be defined with lower-case characters (for example, "debug") or upper-case characters (for example, "DEBUG"). An error occurs if you set a logging level incorrectly, for example, using lower-case characters "debug" where the application defines the logging level as "DEBUG".

## Related Information

[Routing Configuration File \[page 420\]](#)

[Application Router \[page 409\]](#)

[Resource Files \[page 413\]](#)

## 4.1.4.2.2.4 Routing Configuration File

The routing configuration defined in the `xs-app.json` file contains the properties used by the application router.

The following example of an `xs-app.json` application descriptor shows the JSON-compliant syntax required and the properties that either must be set or can be specified as an additional option.

### ↪ Code Syntax

```
{
  "welcomeFile": "index.html",
  "authenticationMethod": "route",
  "sessionTimeout": 10,
  "pluginMetadataEndpoint": "/metadata",
  "routes": [
    {
      "source": "^/sap/ui5/1(.*)$",
      "target": "$1",
      "destination": "ui5",
      "csrfProtection": false
    },
    {
      "source": "/employeeData/(.*)",
      "target": "/services/employeeService/$1",
      "destination": "Employee"
    }
  ]
}
```

```

        "destination": "employeeServices",
        "authenticationType": "xsuaa",
        "scope": ["$XSAPPNAME.viewer", "$XSAPPNAME.writer"],
        "csrfProtection": true
    },
    {
        "source": "^/(.*)$",
        "target": "/web/$1",
        "localDir": "static-content",
        "replace": {
            "pathSuffixes": ["/abc/index.html"],
            "vars": [ "NAME" ]
        },
        {
            "source": "^/user-api(.*)",
            "target": "$1",
            "service": "sap-approuter-userapi"
        }
    ],
    "login": {
        "callbackEndpoint": "/custom/login/callback"
    },
    "logout": {
        "logoutEndpoint": "/my/logout",
        "logoutPage": "/logout-page.html"
    },
    "destinations": {
        "employeeServices": {
            "logoutPath": "/services/employeeService/logout",
            "logoutMethod": "GET"
        }
    },
    "responseHeaders" : [
        { "name": "Content-Security-Policy", "value": "default-src 'self'"}
    ],
    "compression": {
        "minSize": 2048
    },
    "whitelistService": {
        "endpoint": "/whitelist/service"
    },
    "websockets": {
        "enabled": true
    },
    "errorPage": [
        { "status": [400,401,402], "file": "/custom-err-4xx.html" },
        { "status": 501, "file": "/custom-err-501.html" }
    ]
}

```

The following table lists the properties that either must be set or can be specified as an additional option. Click on the links for information for each property:

Property	Type	Description
<a href="#">welcomeFile</a> [page 423]	String	The Web page served by default if the HTTP request does not include a specific path, for example, index.html.
<a href="#">authenticationMethod</a> [page 424]	String	The method used to authenticate user requests, for example: "route" or "none" (no authentication).

Property	Type	Description
<a href="#">sessionTimeout [page 424]</a>	Number	<p>Define the amount of time (in minutes) for which a session can remain inactive before it closes automatically (times out); the default time out is 15 minutes.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>We recommend using the environment variable <a href="#">SESSION_TIMEOUT [page 470]</a> to configure the session timeout.</p> <p>If the environment variable <a href="#">SESSION_TIMEOUT [page 470]</a> is set this property will be overwritten.</p> </div>
<a href="#">routes [page 425]</a>	Array	<p>Defines all route objects, for example: <code>source</code>, <code>target</code>, and, <code>destination</code>.</p>
<a href="#">login [page 438]</a>	Object	<p>A redirect to the application router at a specific endpoint takes place during OAuth2 authentication with the User Account and Authentication service (UAA).</p>
<a href="#">logout [page 438]</a>	Object	<p>You can define any options that apply if you want your application to have central log out end point.</p>
<a href="#">destinations [page 442]</a>	Object	<p>Specify any additional options for your destinations.</p>
<a href="#">services [page 443]</a>	Object	<p>Specify options for a service in your application.</p>
<a href="#">responseHeaders [page 444]</a>	Array	<p>Add custom response headers to your application.</p>
<a href="#">compression [page 445]</a>	Object	<p>The <code>compression</code> keyword enables you to define if the application router compresses text resources before sending them.</p>

Property	Type	Description
<a href="#">pluginMetadataEndpoint [page 446]</a>	String	Adds an endpoint that serves a JSON string representing all configured plugins.
<a href="#">whitelistService [page 446]</a>	Object	Enable the allowlist service to help preventing click-jacking attacks.
<a href="#">websockets [page 448]</a>	Object	The application router can forward web-socket communication. Web-socket communication must be enabled in the application router configuration.
<a href="#">errorPage [page 448]</a>	Array	Errors originating in the application router show the HTTP status code of the error. It is possible to display a custom error page using the <code>errorPage</code> property.

## Related Information

[Application Router \[page 409\]](#)

[Application Router Configuration \[page 414\]](#)

### 4.1.4.2.2.4.1 welcomeFile

The Web page served by default if the HTTP request does not include a specific path, for example, `index.html`.

XS Advanced Application-Router Parameters

Property	Type	Mandatory	Values
<code>welcomeFile</code>	String	No	HTML page, for example, <code>index.html</code>

#### Code Syntax

```
"welcomeFile": "index.html"
```

## 4.1.4.2.2.4.2 authenticationMethod

The method used to authenticate user requests, for example: “route” or “none” (no authentication).

### ↔ Code Syntax

```
"authenticationMethod" : "route"
```

Property	Type	Mandatory	Values
authenticationMethod	String	No	<ul style="list-style-type: none"><li>• route (default) Authentication type is defined in the routes configuration</li><li>• none Disables authentication for all routes</li></ul>

### ⚠ Caution

If authenticationMethod is set to “none”, logon with User Account and Authentication (UAA) is disabled.

## 4.1.4.2.2.4.3 sessionTimeout

Define the amount of time (in minutes) for which a session can remain inactive before it closes automatically (times out); the default time out is 15 minutes.

### ⓘ Note

We recommend using the environment variable [SESSION\\_TIMEOUT \[page 470\]](#) to configure the session timeout.

If the environment variable [SESSION\\_TIMEOUT \[page 470\]](#) is set this property will be overwritten.

### ↔ Sample Code

```
{  
  "sessionTimeout": 40  
}
```

With the configuration in the example above, a session timeout will be triggered after 40 minutes and involves central log out.

A session timeout triggers a central log out with the following consequences:

- Deletes the user session
- Requests the log out paths for all your back-end services (if you provided these paths in the destinations and service properties).

#### ⓘ Note

sessionTimeout can only be configured in the central routing configuration file (xs-app.json) that belongs to the application router, not in an xs-app.json file that is part of an HTML5 application.

### 4.1.4.2.2.4.4 routes

Defines all route objects, for example: source, target, and, destination.

- [Routes Properties \[page 425\]](#)
- [Route Configuration Examples \[page 432\]](#)

Application Router: Routes Properties

Property	Type	Mandatory	Description
source	RegEx	Yes	<p>Describes a regular expression that matches the incoming request URL.</p> <p>A request matches a particular route when its path contains the given pattern. To ensure the RegEx matches the complete path, use the following form: `^\$`.</p>
httpMethods <a href="#">[page 429]</a>	Array of uppercase HTTP methods	No	<p>HTTP methods that are served by this route; the supported methods are: DELETE, GET, HEAD, OPTIONS, POST, PUT, TRACE, and PATCH.</p> <p>→ Tip</p> <p>If this option isn't specified, the route serves any HTTP method.</p>
target	String	No	Defines how the incoming request path is rewritten for the corresponding destination or static resource.

Property	Type	Mandatory	Description
destination	String	No	<p>The name of the destination to which the incoming request is forwarded. The destination name can be a static string or a regular expression that defines how to dynamically fetch the destination name from the source property or from the host.</p> <p>For more information about additional destination properties, see <a href="#">Application Routes and Destinations [page 450]</a>.</p>
service	String	No	The name of the service to which the incoming request is forwarded.
endpoint	String	No	The name of the endpoint within the service to which the incoming request is forwarded. It must only be used in a route containing a service attribute.
<a href="#">localDir [page 430]</a>	String	No	The directory from which application router serves static content (for example, from the application's web/ module).
preferLocal	Boolean	No	Defines from which subaccount the destination is retrieved. If preferLocal is true, the destination is retrieved from the provider subaccount. If preferLocal is false or undefined, the destination is retrieved from the subscriber subaccount.
<a href="#">replace [page 430]</a>	Object	No	An object that contains the configuration for replacing placeholders with values from the environment.

#### ⓘ Note

It is only relevant for static content.

Property	Type	Mandatory	Description
authenticationType	String	No	<p>The value can be <code>ias</code>, <code>xsuaa</code>, <code>basic</code>, or <code>none</code>.</p> <p>The default <code>authenticationType</code> depends on the authentication service binding: If the application router is bound to the Identity Authentication service, the default <code>authenticationType</code> is <code>ias</code>. Otherwise, the default value is <code>xsuaa</code>.</p> <p>If you use the value <code>xsuaa</code> or <code>ias</code>, the specified authentication server (Identity Authentication or User Account and Authentication) handles the authentication and the user is redirected to the login form of Identity Authentication or User Account and Authentication.</p> <p>The <code>basic</code> authenticationType works with SAP S/4 HANA users, SAP ID service, and Identity Authentication service. For more information, see the SAP Note 3015211 - BASIC authentication options for SAP BTP Cloud Foundry applications.</p> <p>If the value <code>none</code> is used, no authentication is required for this route.</p>
csrfProtection	Boolean	No	<p>Toggle whether this route needs CSRF token protection. The default value is "true". The application router enforces CSRF protection for any HTTP request that changes state on the server side, for example: PUT, POST, or DELETE.</p>
scope	Array/String/Object	No	<p>The authorization scope required to access the target path. The scope itself is defined in the application's security descriptor (<code>xs-security.json</code>), for example, "<code>XSAPPNAME.Display</code>" or "<code>XSAPPNAME.Create</code>".</p>
cacheControl	String	No	<p>A string representing the value of the Cache-Control header, which is set on the response when serving static resources. By default the Cache-Control header isn't set.</p> <p>The cacheControl property is only effective when one of the following settings is performed:</p> <ul style="list-style-type: none"> <li>• localDir property is set</li> <li>• Service pointing to HTML5 Application Repository ("service": "html5-apps-repo-rt") is set</li> </ul>

Property	Type	Mandatory	Description
identityProvider	String	No	The name of the identity provider you use if it's provided in the definition of the route. If it isn't provided, the route is authenticated with the default identity provider.
dynamicIdentityProvider	Boolean	No	<p>Enables dynamic identity provider provisioning.</p> <p>If <code>dynamicIdentityProvider</code> is <code>true</code>, the end user can set the identity provider (IDP) for the application's login process by filling the request query parameter <code>sap_idp</code> with the IDP origin key. If the <code>IdentityProvider</code> property is defined in the route, its value is overwritten by the <code>sap_idp</code> query parameter value. The default value for <code>dynamicIdentityProvider</code> is <code>false</code>.</p>

## ⓘ Note

Route order is important. The first matching route will be used. Therefore, it is recommended to sort the routes from the most specific source to the more generic one. For example:

### ↪ Sample Code

```
"routes": [
  {
    "source": "^/sap/backend/employees(.*)$",
    "target": "$1",
    "destination": "sfsf"
  },
  {
    "source": "^/sap/backend/(.*)$",
    "target": "$1",
    "destination": "erp",
  }
]
```

### ↪ Code Syntax

```
"routes": [
  {
    "source": "^/sap/ui5/1(.*)$",
    "target": "$1",
    "destination": "ui5",
    "scope": "$XSAPPNAME.viewer",
    "authenticationType": "xsuaa",
  }
]
```

```
        "csrfProtection": true
    }
]
```

## ⓘ Note

The properties `service`, `destination`, and `localDir` are optional. However, at least one of them **must** be defined.

### httpMethods

The `httpMethods` option allows you to split the same path across different targets depending on the HTTP method. For example:

#### ↗ Sample Code

```
"routes": [
{
    "source": "^/app1/(.*)$",
    "target": "/before/$1/after",
    "httpMethods": [ "GET", "POST" ]
}
]
```

This route only serves GET and POST requests. Any other method (including extension ones) gets a 405 Method Not Allowed response. The same endpoint can be split across multiple destinations depending on the HTTP method of the requests:

#### ↗ Sample Code

```
"routes": [
{
    "source": "^/app1/(.*)$",
    "destination" : "dest-1",
    "httpMethods": [ "GET" ]
},
{
    "source": "^/app1/(.*)$",
    "destination" : "dest-2",
    "httpMethods": [ "DELETE", "POST", "PUT" ]
}
]
```

This sample code routes GET requests to the target `dest-1`, DELETE, POST and PUT to `dest-2`, and any other method receives a 405 Method Not Allowed response. It's also possible to specify `catchAll` routes, namely routes that don't specify `httpMethods` restrictions:

#### ↗ Sample Code

```
"routes": [
{
    "source": "^/app1/(.*)$",
    "destination" : "dest-1",
}
```

```
        "httpMethods": [ "GET" ]
    },
{
    "source": "^/app1/(.*)$",
    "destination": "dest-2"
}
]
```

In this sample code, GET requests are routed to dest-1, and all of the rest are routed to dest-2.

## localDir

If there's no route defined for serving static content via `localDir`, a default route is created for "resources" directory as follows:

### ↔ Sample Code

```
{
    "routes": [
        {
            "source": "^/(.*)$",
            "localDir": "resources"
        }
    ]
}
```

### ⓘ Note

If there is at least one route using `localDir`, the default route isn't added.

## replace

The `replace` object configures the placeholder replacement in static text resources.

### ↔ Sample Code

```
{
    "replace": {
        "pathSuffixes": [ "index.html" ],
        "vars": [ "escaped_text", "NOT_ESCAPED" ]
    }
}
```

The `replace` keyword requires the following properties:

Replacement Properties for Static Resource URLs

Property	Type	Description
<code>pathSuffixes</code>	Array	An array defining the path suffixes that are relative to <code>localDir</code> . Only files with a path ending with any of these suffixes are processed.
<code>vars</code>	Array	A whitelist with the environment variables that are replaced in the files matching the suffix specified in <code>pathSuffixes</code> .

The supported tags for replacing environment variables are: `{ { ENV_VAR } }` and `{ { { ENV_VAR } } }`. If such an environment variable is defined, it's replaced; otherwise, it's just an empty string.

### ⓘ Note

Any variable that is replaced using two-brackets syntax `{ { ENV_VAR } }` is HTML-escaped; the triple brackets syntax `{ { { ENV_VAR } } }` is used when the replaced values don't need to be escaped and all values remain unchanged. For example, if the value of the environment variable is `ab" cd` the result is `ab&quot;cd`.

If your application descriptor `xs-app.json` contains a route like the one illustrated in the following example,

```
{
  "source": "^\/get/home(.*)",
  "target": "$1",
  "localDir": "resources",
  "replace": {
    "pathSuffixes": [ "index.html" ],
    "vars": [ "escaped_text", "NOT_ESCAPED" ]
  }
}
```

And your application uses the following `index.html` start file:

### ↴ Sample Code

```
<html>
<head>
  <title>{{escaped_text}}</title>
  <script src="{{{NOT_ESCAPED}}}/index.js"/>
</head>
</html>
```

Then, in the `index.html`, `{{escaped_text}}` and `{{{NOT_ESCAPED}}}` are replaced with the value defined in the environment variables `<escaped_text>` and `<NOT_ESCAPED>`.

### ⓘ Note

All `index.html` files are processed; if you want to apply the replacement only to specific files, you must set the path relative to `localDir`. In addition, all files must comply with the UTF-8 encoding rules.

The content type returned by a request is based on the file extension specified in the route. The application router supports the following file types:

- .json (application/json)
- .txt (text/plain)
- .html (text/html) **default**
- .js (application/javascript)
- .css (text/css)

The following table illustrates some examples of the `pathSuffixes` properties:

Examples of the `pathSuffixes` Property

Example	Result
{ "pathSuffixes": [ ".html" ] }	All files with the extension <code>.html</code> under <code>localDir</code> and its subfolders are processed.
{ "pathSuffixes": [ "/abc/main.html", "some.html" ] }	For the suffix <code>/abc/main.html</code> , all files named <code>main.html</code> that are inside a folder named <code>abc</code> are processed.  For the suffix <code>some.html</code> , all files with a name that ends with " <code>some.html</code> " are processed. For example: <code>some.html</code> , <code>awesome.html</code> .
{ "pathSuffixes": [ ".html" ] }	All files with the name " <code>some.html</code> " are processed. For example: <code>some.html</code> , <code>/abc/some.html</code> .

## Route Configuration Examples

### Route with a destination and no target

↔ Sample Code

```
{  
  "source": "^/app1/(.*)$"  
  "destination": "app-1"  
}
```

Since there is no target property for that route, no path rewriting will take place. If `/app1/a/b` is received as a path, then a request to `http://localhost:3001/app1/a/b` is sent. The source path is appended to the destination URL.

### Route with case-insensitive matching

↔ Sample Code

```
{  
  "source": {  
    "path": "^/app1/(.*)$"  
    "matchCase": false  
  },  
  "destination": "app-1"
```

```
}
```

### ⓘ Note

The property `matchCase` must be boolean. It is optional and has a default value of `true`.

## Route with a destination and a target

### ↗ Sample Code

```
{
  "source": "^/app1/(.*)$",
  "target": "/before/$1/after",
  "destination": "app-1"
}
```

## Route with a service, a target and an endpoint

### ↗ Sample Code

```
{
  "source": "^/odata/v2/(.*)$",
  "target": "$1",
  "service": "com.sap.appbasic.country",
  "endpoint": "countryservice"
}
```

When a request with path `/app1/a/b` is received, the path rewriting is done according to the rules in the target property. The request will be forwarded to `http://localhost:3001/before/a/b/after`.

### ⓘ Note

In regular expressions there is the term capturing group. If a part of a regular expression is surrounded with parenthesis, then what has been matched can be accessed using `$ + the number of the group (starting from 1)`. In code sample, `$1` is mapped to the `(.*)` part of the regular expression in the source property.

## Route with dynamic destination and target

### ↗ Sample Code

```
{
  "source": "^/destination/([^\n]+)/(.*$)",
  "target": "$2",
  "destination": "$1",
  "authenticationType": "xsuaa"
}
```

If you have another destination configured, use this:

### ↗ Sample Code

```
[
  {
    "name" : "myDestination",
```

```
        "url" : "http://localhost:3002"
    }
]
```

When a request with the path `/destination/myDestination/myTarget` is received, the destination will be replaced with the URL from "myDestination", the target will get "myTarget" and the request will be redirected to `http://localhost:3002/myTarget`.

#### ⓘ Note

You can use a dynamic value (regex) or a static string for `destination` and `target` values.

The Application Router first looks for the destination name in the `manifest.yaml` file, and if it is not found, it looks for it in the destination service.

## Destination in Host

For legacy applications that do not support relative URL paths, you need to define your URL in the following way to enable the destination to be extracted from the host:

```
https://<tenant>-<destination>.<customdomain>/<pathoffile>
```

To enable the application router to determine the destination of the URL host, a `DESTINATION_HOST_PATTERN` attribute must be provided as an environment variable. For example, When a request with the path `https://myDestination.some-approuter.someDomain.com/app1/myTarget` is received, the following route is used:

#### ↴ Sample Code

```
{
  "source": "^/app1/([^\n]+)\/",
  "target": "$1",
  "destination": "*",
  "authenticationType": "xsuaa"
}
```

In this example, the target will be extracted from the source and the '\$1' value is replaced with "myTarget". The destination value is extracted from the host and the "\*" value is replaced with "myDestination".

## Route with a localDir and no target

#### ↴ Sample Code

```
{
  "source": "^/web-pages/(.*)$",
  "target": "$1",
  "localDir": "my-static-resources"
}
```

If we receive a request with a path `/web-pages/welcome-page.html`, the local file at `my-static-resources/welcome-page.html` under the working directory will be served.

#### ⓘ Note

The capturing group used in the `target` property.

## Route with localDir and cacheControl

### ↔ Sample Code

```
{  
  "source": "^/web-pages/",  
  "localDir": "my-static-resources",  
  "cacheControl": "public, max-age=1000,must-revalidate"  
}
```

## Route with service "html5-apps-repo-rt" and cacheControl

### ↔ Sample Code

```
{  
  "source": "^/index.html$",  
  "service": "html5-apps-repo-rt",  
  "authenticationType": "xsuaa",  
  "cacheControl": "public,max-age=1000,must-revalidate"  
}
```

## Route with httpMethods restrictions

This option allows you to split the same path across different targets depending on the HTTP method. For example:

### ↔ Sample Code

```
{  
  "source": "^/app1/(.*)$",  
  "target": "/before/$1/after",  
  "httpMethods": [ "GET", "POST" ]  
}
```

This route only supports GET and POST requests. Any other method (including extensions) will receive a [405 Method Not Allowed](#) response. The same endpoint can be split across multiple destinations depending on the HTTP method of the requests:

### ↔ Sample Code

```
{  
  "source": "^/app1/(.*)$",  
  "destination" : "dest-1",  
  "httpMethods": [ "GET" ]  
},  
{  
  "source": "^/app1/(.*)$",  
  "destination" : "dest-2",  
  "httpMethods": [ "DELETE", "POST", "PUT" ]  
}
```

In the setup above, GET requests will be routed to "dest-1", and all the rest to "dest-2".

### ⓘ Note

localDir and httpMethods are not compatible.

## Route with a scope

An application specific scope uses the following format:

```
<application-name>.<scope-name>
```

It is possible to configure what scope the user needs to possess in order to access a specific resource. Those configurations are per route. The user should have at least one of the scopes in order to access the corresponding resource.

### ↳ Sample Code

```
{  
    "source": "^/web-pages/(.*)$"  
    "target": "$1",  
    "scope": [ "$XSAPPNAME.viewer", "$XSAPPNAME.reader", "$XSAPPNAME.writer" ]  
}
```

For convenience if your route only requires one scope, the scope property can be a string instead of an array. The following configuration is also valid:

### ↳ Sample Code

```
{  
    "source": "^/web-pages/(.*)$"  
    "target": "$1",  
    "scope": "$XSAPPNAME.viewer"  
}
```

You can configure scopes for different HTTP methods, such as GET, POST, PUT, HEAD, DELETE, CONNECT, TRACE, PATCH, andOPTIONS. If some of the HTTP methods are not explicitly set, the behaviour for them is defined by the default property. In case there is no default property specified and the HTTP method is also not specified, the request is rejected by default.

### ↳ Sample Code

```
{  
    "source": "^/web-pages/(.*)$"  
    "target": "$1",  
    "scope": {  
        "GET": "$XSAPPNAME.viewer",  
        "POST": [ "$XSAPPNAME.reader", "$XSAPPNAME.writer" ],  
        "default": "$XSAPPNAME.guest"  
    }  
}
```

The application router supports the \$XSAPPNAME placeholder. Its value is taken (and then substituted in the routes) from the UAA configuration.

### ⓘ Note

The substitution is case sensitive.

You can use the name of the business application directly instead of using the \$XSAPPNAME placeholder:

### ↳ Sample Code

```
{
```

```
        "source": "^/backend/(.*)$",
        "scope": "my-business-application.viewer"
    }
```

## Routes with an `identityProvider`

You can define several identity providers for different types of users. In this code example, there are two categories: hospital patients and hospital personnel:

- `patientsIDP` – use for authenticating patients.
- `hospitalIDP` – use for authenticating all hospital personnel (doctors, nurses etc..).

### ↔ Sample Code

```
[
  {
    "source": "^/patients/sap/opu/odata/(.*)",
    "target": "/sap/opu/odata$1",
    "destination": "backend",
    "authenticationType": "xsuaa",
    "identityProvider": "patientsIDP"
  },
  {
    "source": "^/hospital/sap/opu/odata/(.*)",
    "target": "/sap/opu/odata$1",
    "destination": "backend", "authenticationType": "xsuaa",
    "identityProvider": "hospitalIDP"
  }
]
```

A patient who tries to log into the system will be authenticated by `patientIDP`, and a doctor who tries to log in will be authenticated by `hospitalIDP`.

### ⓘ Note

If a user logs in as one identity and then wants to perform tasks for another identity type, the user must log out and log back in to the system.

Dynamic provisioning of the subscriber account identity provider is not supported.

Identity provider configuration is only supported in the client side logon redirect flow.

## Route with a `dynamicIdentityProvider`

This is an example of a route where the value of `identityProvider` is `patientsIDP` and where dynamic identity provider provisioning is enabled by setting `dynamicIdentityProvider` to true:

### ↔ Sample Code

```
[
  {
    "source": "^/patients/index.html",
    "target": "/patients-index.html",
    "service": "html5-apps-repo-rt",
    "identityProvider": "patientsIDP",
    "dynamicIdentityProvider": true
  }
]
```

In this example, the `patientsIDP` value for the `identityProvider` is replaced by `hospitalIDP` if a request with `sap_idp=hospitalIDP` is executed, for example, if the request is `https://shiva.health-center-approuter.cfapps.hana.ondemand.com/healthreport/patients/index.html?sap_idp=hospitalIDP`.

#### 4.1.4.2.2.4.5 login

A redirect to the application router at a specific endpoint takes place during OAuth2 authentication with the service that you are using for the authentication (User Account and Authentication service (UAA) or Identity Authentication service).

This endpoint can be configured in order to avoid possible collisions, as illustrated in the following example:

##### ↔ Sample Code

Application Router “login” Property

```
"login": {  
    "callbackEndpoint": "/custom/login/callback"  
}
```

##### → Tip

The default endpoint is “/login/callback”.

#### 4.1.4.2.2.4.6 logout

You can define any options that apply if you want your application to have a central logout end point.

In this object you can define an application's central logout end point by using the `logoutEndpoint` property. The value of `logout` property should be an object with the following properties:

Property	Type	Mandatory	Description
<code>logoutEndpoint</code>	String	Yes	The path to be used when logging out from the application router
<code>logoutPage</code>	String	No	The URL path of the logout page

Property	Type	Mandatory	Description
logoutMethod	String	No	<p>Can be POST or GET. The default value is GET.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <b>① Note</b>            For security reasons, it is recommended to use the POST method and to enable CSRF protection.         </div>
csrfProtection	Boolean	No	<p>Can only be defined if the logoutMethod is POST. If the logoutMethod is POST and this property is not defined, the default value is true. You can set it to false – for example if csrfProtection is implemented in back-end application</p>

This is an example:

#### ↔ Sample Code

```
"logout" {
  "logoutEndpoint": "/my/logout"
}
```

Making a GET or POST request to “/my/logout” triggers a client-initiated central logout with the following consequences:

- Deletes the user session
- Requests the logout paths for all your back-end services (if you provided these paths in the destinations and service properties).
- Redirects to the authentication service (UAA or Identity Authentication - depending on which service you are using for the authentication), if such a service is provided, and logs out from there.

You can use the `logoutPage` property to specify the Web page in one of the following ways:

- URL path  
The authentication service (UAA or Identity Authentication - depending on which service you are using for the authentication) redirects the user back to the application router, and the path is interpreted according to the configured routes.  
The `logoutEndpoint` can be called with the `skip-redirect` or `skip-redirect=true` query parameter to prevent the application router from redirecting to the Identity Authentication service (IAS) for logout after ending the application router user session. Use this configuration if the logout process is initiated by Identity Authentication . In this case, the end user initiates the single logout flow of Identity

Authentication, and subsequently, the application router logout endpoint is called. The application router logout endpoint must not redirect to the Identity Authentication logout endpoint in this case. For more details see [Handle Single Logout Request from Identity Authentication](#). For example:

#### ↔ Sample Code

```
window.location.replace('/my/logout?skip-redirect')
```

The `logoutEndpoint` can be called with query parameters. For example:

#### ↔ Sample Code

```
window.location.replace('/my/logout?siteId=3');
```

These parameters will be appended as is to the redirect URL set by the `logoutPage` property. For example, if the logout section is:

#### ↔ Sample Code

```
"logout": {  
    "logoutEndpoint": "/logout",  
    "logoutPage": "/logoff.html"  
},
```

The redirect URL will end with:

#### ↔ Sample Code

```
/logoff.html?siteId=3
```

#### ⓘ Note

The resource that matches the URL path specified in the property `logoutPage` should not require authentication; for this route, the property `authenticationType` must be set to `none`.

In the following example, `my-static-resources` is a folder in the working directory of the application router; the folder contains the file `logout-page.html` along with other static resources.

#### ↔ Sample Code

```
{  
    "authenticationMethod": "route",  
    "logout": {  
        "logoutEndpoint": "/my/logout",  
        "logoutPage": "/logout-page.html"  
    },  
    "routes": [  
        {  
            "source": "^/logout-page.html$",  
            "localDir": "my-static-resources",  
            "authenticationType": "none"  
        }  
    ]  
}
```

- Absolute HTTP(S) path

The authentication service (UAA or Identity Authentication - depending on which service you are using for the authentication) redirects the user to a page (or application) different from the application router.

#### ↔ Sample Code

```
"logout": {
  "logoutEndpoint": "/my/logout",
  "logoutPage": "http://acme.com/employees.portal"
}
```

## Using the POST Method for Logout

For security reasons, it is recommended to use [POST](#) method for logout and to enable CSRF protection.

The `logoutMethod` and `csrfProtection` properties are included in the `logout` property:

#### ↔ Sample Code

```
"logout": {
  "logoutEndpoint": "/my/logout",
  "logoutPage": "/logout-page.html",
  "logoutMethod": "POST",
  "csrfProtection": true
}
```

#### ⓘ Note

For backward compatibility reasons, the default value of `logoutMethod` is `GET`. The `csrfProtection` property can only be set if the `logoutMethod` is `POST`. If the `logoutMethod` is `POST` and the `csrfProtection` property is not set, the `csrfProtection` is enabled by default.

In this example, the `POST` request is an AJAX request and includes a CSRF token:

#### ↔ Sample Code

```
async function getToken() {
  return new Promise((resolve) => {
    jQuery.ajax({
      type: "GET",
      url: 'my/logout',
      headers: {
        "X-CSRF-Token": 'fetch',
        contentType: "application/json",
      },
      success: function(data, textStatus, request){
        resolve(request.getResponseHeader('X-CSRF-Token'));
      },
    });
  });
}
```

This is an example for the `POST` request:

#### ↔ Sample Code

```
const token = await getToken();
jQuery.ajax({
  type: "POST",
  url: "my/logout",
  headers: {
    "X-CSRF-Token": token,
    "contentType": "application/json",
  },
  success: function (data) {
    window.location.href = data;
  }
});
```

#### ⓘ Note

Make sure that the `url` field matches the `logoutEndpoint`.

### 4.1.4.2.2.4.7 destinations

Specify any additional options for your destinations.

The destinations section in `xs-app.json` extends the destination configuration in the deployment manifest (`manifest.yml`), for example, with some static properties such as a logout path.

#### ↔ Sample Code

```
{
  "destinations": {
    "node-backend": {
      "logoutPath": "/ui5logout",
      "logoutMethod": "GET"
    }
  }
}
```

The following syntax rules apply:

Application Router: Destination Properties

Property	Type	Mandatory	Description
<code>logoutPath</code>	String	No	The log out end point for your destination. The <code>logoutPath</code> will be called when central log out is triggered or a session is deleted due to a time out. The request to <code>logoutPath</code> contains additional headers, including the JWT token.

Property	Type	Mandatory	Description
logoutMethod	String	No	The <code>logoutMethod</code> property specifies the HTTP method with which the <code>logoutPath</code> will be requested, for example, POST, PUT, GET; the default value is POST

## 4.1.4.2.2.4.8 services

Specify options for a service in your application.

The services section in `xs-app.json` extends the services configuration in the deployment manifest (`manifest.yml`), for example, with some static properties such as an end point.

### Sample Code

```
{
  "services": {
    "com.sap.appbasic.country": {
      "endpoint": "countrieservice",
      "logoutPath": "/countrieslogout",
      "logoutMethod": "GET"
    }
  }
}
```

The following syntax rules apply:

Application Router: Services Properties

Property	Type	Mandatory	Description
endpoint	String	No	The name of the attribute in the <code>VCAP_SERVICES</code> that contains the URL of the service.
logoutPath	String	No	The log out end point for your destination. The <code>logoutPath</code> will be called when central log out is triggered or a session is deleted due to a time out. The request to <code>logoutPath</code> contains additional headers, including the JWT token.
logoutMethod	String	No	The <code>logoutMethod</code> property specifies the HTTP method with which the <code>logoutPath</code> will be requested, for example, POST, PUT, GET; the default value is POST

## Related Information

[Integration with Business Services \[page 459\]](#)

### 4.1.4.2.2.4.9 responseHeaders

You can add headers that the application router returns to the client in its responses.

You can add response headers to your application, for example, to comply with security standards.

The `responseHeaders` property is an array of objects, each object having the following properties:

Property	Type	Description
name	string	response header name
value	string	response header value

```
{ "responseHeaders" : [
    { "name": "header1", "value": "value1" },
    { "name": "header2", "value": "value2" }
]}
```

Example:

#### ↔ Sample Code

```
{ "responseHeaders" : [
    { "name": "Content-Security-Policy", "value": "default-src 'self'" }
]}
```

#### ⓘ Note

The response headers are added to the list of additional http headers that you have configured in the `<httpHeaders>` environment variable [\[page 463\]](#). If the name of a response header configured in the routing configuration is identical with name of an additional http header in the list, the value of the response header overrides the value of the additional http header.

## Related Information

[Environment Variables \[page 463\]](#)

## 4.1.4.2.2.4.10 compression

The `compression` keyword enables you to define if the application router compresses text resources before sending them.

By default, resources larger than 1KB are compressed. If you need to change the compression size threshold, for example, to “2048 bytes”, you can add the optional property “`minSize`”: `<size_in_KB>`, as illustrated in the following example.

### ↔ Sample Code

```
{  
    "compression": {  
        "minSize": 2048  
    }  
}
```

You can disable compression in the following ways:

- Global  
Within the compression section add “`enabled`”: `false`
- Front end  
The client sends a header “`Accept-Encoding`” which does not include “`gzip`”.
- Back end  
The application sends a header “`Cache-Control`” with the “`no-transform`” directive.

Application Router: Compression Properties

Property	Type	Mandatory	Description
<code>minSize</code>	Number	No	Text resources larger than this size will be compressed.
<code>enabled</code>	Boolean	No	Globally disables or enables compression. The default value is <code>true</code> .
<code>compressResponseMix</code> <code>edTypeContent</code>	Boolean	No	Determines whether response content of the multipart / mixed content type should be compressed. The default value is <code>false</code> .

### ⓘ Note

If the `<COMPRESSION>` environment variable is set it will overwrite any existing values.

## 4.1.4.2.2.4.11 pluginMetadataEndpoint

Adds an endpoint that serves a JSON string representing all configured plugins.

### ↳ Sample Code

```
{  
  "pluginMetadataEndpoint": "/metadata"  
}
```

### ⓘ Note

If you request the relative path `/metadata` of your application, a JSON string is returned with the configured plug-ins.

## 4.1.4.2.2.4.12 whitelistService

Enable the allowlist service to help prevent against click-jacking attacks.

Enabling the allowlist service (which is called `whitelistService` in the code) opens an endpoint accepting GET requests at the relative path configured in the `endpoint` property, as illustrated in the following example:

### ↳ Sample Code

```
{  
  "whitelistService": {  
    "endpoint": "/whitelist/service"  
  }  
}
```

If the allowlist service is enabled in the application router, each time an HTML page needs to be rendered in a frame, the allowlist service is used check if the parent frame is allowed to render the content in a frame.

### Host Names and Domain Names

The allowlist service reads a list of allowed host names and domains defined in the environment variable `<CU_PROTECT_WHITELIST>`; the content is a JSON list of objects with the following properties:

Allowlist of Host and Domain Names

Property	Type	Mandatory	Description
protocol	String	No	URI scheme, for example "HTTP".
host	String	Yes	A valid host name, for example, <code>acme.com.hostname</code> , or a domain name defined with an asterisk (*) <code>*.acme.com</code> .
port	String/Number	No	Port string or number containing a valid port.

The following snippet shows an example of the resulting JSON array:

#### ↔ Sample Code

```
[  
  {  
    "protocol": "http",  
    "host": "*.acme.com",  
    "port": 12345  
  },  
  {  
    "host": "hostname.acme.com"  
  }]
```

#### ⓘ Note

Matching is done against the properties provided. For example, if only host name is provided, the allowlist service returns “framing: true” for all, and matching will be for all schemata and protocols.

#### Return Value

The allowlist service accepts only GET requests, and returns a JSON object as the response. The allowlist service call uses the parent origin as URI parameter (URL encoded) as follows:

#### ↔ Sample Code

```
GET url/to/whitelist/service?parentOrigin=https://parent.domain.acme.com
```

The response is a JSON object with the following properties; property “active” has the value false only if <CU\_PROTECT\_WHITELIST> is not provided:

#### ↔ Sample Code

```
{  
  "version" : "1.0",  
  "active" : true | false,  
  "origin" : "<same as passed to service>",  
  "framing" : true | false  
}
```

The “active” property enables framing control; the “framing” property specifies if framing should be allowed. By default, the application router (approuter.js) sends the X-Frame-Options header with value the SAMEORIGIN.

#### → Tip

If the allowlist service is enabled, the header value probably needs to be changed, see the X-Frame-Options header section for details about how to change it.

## 4.1.4.2.2.4.13 websockets

The application router can forward web-socket communication. Web-socket communication must be enabled in the application router configuration.

If the back-end service requires authentication, the upgrade request should contain a valid session cookie. The application router supports the destination schemata "ws", "wss", "http", and "https".

### ↔ Sample Code

```
{  
  "websockets": {  
    "enabled": true  
  }  
}
```

To use web-sockets when the application router is integrated with the HTML5 Application Repository, the `websockets` property should be added to the `xs-app.json` of the deployed HTML5 application. When an incoming request for an application in the repository goes through the application router, it retrieves the application's configuration from the repository. If this flag is set, the application router creates a web-socket connection to the back end (the target url of the request) and acts as a proxy which delivers messages on top of the `ws` protocol from the back end to the user, and vice versa.

### ⚠ Restriction

A web-socket ping is not forwarded to the back-end service.

## 4.1.4.2.2.4.14 errorPage

Errors originating in the application router show the HTTP status code of the error. It is possible to display a custom error page using the `errorPage` property.

The property is an array of objects, each object can have the following properties:

Application Router: `errorPage` Properties

Property	Type	Mandatory	Description
<code>status</code>	Number/Array	Yes	HTTP status code.
<code>file</code>	String	Yes	File path relative to the working directory of the application router.

In the following code example, errors with status code “400”, “401” and “402” will show the content of `./custom-err-4xx.html`; errors with the status code “501” will display the content of `./http_resources/custom-err-501.html`.

### ↔ Sample Code

```
{ "errorPage" : [
```

```

        {"status": [400,401,402], "file": "./custom-err-40x.html"},  

        {"status": 501, "file": "./http_resources/custom-err-501.html"}  

    ]  

}

```

### ① Note

The contents of the `errorPage` configuration section have no effect on errors that are not generated by the application router.

## 4.1.4.2.2.5 Headers

### Forwarding Header

The application router sends the following `x-forwarding-` headers to the route targets:

Header Name	Description
<code>x-forwarded-host</code>	Contains the host header that is sent from the client to the application router.
<code>x-forwarded-proto</code>	Contains the protocol that is used by the client to connect to the application router.
<code>x-forwarded-for</code>	Contains the address of the client that connects to the application router.
<code>x-forwarded-path</code>	Contains the original path that the client requested.

If a client performs a path rewriting, it sends the `x-forwarded-proto`, `x-forwarded-host`, and the `x-forwarded-path` headers to the application router. The values of these headers are forwarded to the route targets without modifications instead of being generated from the application router request URL. The `x-forwarded-path` header of a request does not impact the source pattern of routes in the `xs-app.json`.

### Hop-by-Hop Headers

Hop-by-hop headers are only for a single transport-level connection and are not forwarded by the application router. The headers are:

- `Connection`
- `Keep-Alive`
- `Public`
- `Proxy-Authenticate`
- `Transfer-Encoding`

- Upgrade

## Custom Header

`x-custom-host` is used to support the application router behind an external reverse proxy. The `x-custom-host` header must contain the internal reverse proxy host.

### ⓘ Note

`EXTERNAL_REVERSE_PROXY` environment variable is set to true.

In a multi-tenancy landscape, application router can be called from multiple tenants. During the authentication flow, application router uses the tenant ID to fetch the authentication token from XSUAA. Application router extracts the tenant ID from the corresponding host using the tenant host pattern configuration.

In an external reverse proxy flow, the application router uses the `x-custom-host` to extract the tenant ID using the tenant host pattern configuration.

If the `x-custom-host` is not provided, the application router uses the host header to extract the tenant ID.

## Authorization Header for Service to Application Router

The `x-approuter-authorization` header contains the JWT token or Open ID Connect (OIDC) access token to support the service-to-application-router scenario.

The application router can receive a JWT token created by the SAP Authorization and Trust Management service (XSUAA) service or an OIDC access token created by Identity Authentication and use it to access the UI and the data. The JWT token is passed to the application router in the `x-approuter-authorization` header of a request.

### ⓘ Note

The JWT token or the OIDC token are generated with the same SAP Authorization and Trust Management service (XSUAA) service instance or the same Identity Authentication service instance that is bound to the application router.

## 4.1.4.2.2.6 Application Routes and Destinations

The application router is the single point of entry for an application.

The application router is used to serve static content, propagates user information, and acts as a proxy to forward requests to other microservices. The routing configuration for an application is defined in one or more destinations. The application router configuration is responsible for the following tasks:

- Act as the main user-entry point to application service

- Serve static content
- Serve routes and destinations
- Rewrite URLs
- Forward requests to other services
- Propagate user information
- Handle authentication
- Manage HTML5 application/browser sessions

#### *(i)* Note

The application router doesn't manage server caching. The server cache must be set (with e-tags) in the server itself. The cache must contain not only static content, but container resources too. For example, relating to OData metadata.

## Destinations

A destination defines the back-end connectivity. In its simplest form, a destination is a URL to which requests are forwarded. There has to be a destination for every single app (microservice) that is a part of the business application.

#### *(i)* Note

You must not use multiple destinations with the same URL in an application router user session!

The application router stores the session cookies from the backend services in a user session according to the destination URLs for the backend services. If there are multiple destinations with the same URL, the application router cannot correctly send back the cookies to the backend services. Note that to avoid conflicts, the destination URLs must have different domains. For more information, see [Sessions \[page 416\]](#).

The destinations configuration can be provided by the destinations environment variable or by the destination service.

### Destinations Environment Variable

Destinations can be defined in an environment variable for the `approuter` application. The destinations are typically specified in the application manifest file (`manifest.yml`). The router information is added to the `env: destinations` section of the `manifest.yml` file, as illustrated in the following example:

#### *↔* Sample Code

Destinations Defined in the Application Manifest (`manifest.yml`)

```
---
applications:
- name: node-hello-world
  port: <approuter-port> #the port used for the approuter
  memory: 100M
  path: web
  env:
    destinations: >
      [
        {
          name: <destination-name>
          url: <destination-url>
          protocol: <protocol>
          port: <port>
          auth: <auth>
          headers: <headers>
        }
      ]
  
```

```

        {
          "name": "backend",
          "url": "http://<hostname>:<node-port>",
          "forwardAuthToken": true
        }
      ]
    services:
      - node-uaa
  
```

### Note

The "name" and "url" properties are mandatory.

The value of the destination "name" : "backend" must match the value of the `destinations` property configured for a route in the corresponding application-router configuration file (`xs-app.json`). It's also possible to define a logout path and method for the `destinations` property in the `xs-app.json` file.

## Destination Service

Destination configuration can be provided by `destination service`. The destination service can be consumed by the application router after binding a destination service instance to it.

The following guidelines apply to the configuration of the mandatory properties of a destination:

Mandatory Properties of a Destination

Property	Description
Type	Only HTTP is supported.
Authentication	All authentication types are supported.
ProxyType	<p>Supported types:</p> <ul style="list-style-type: none"> <li>• <code>on-premise</code> If set, binding to SAP Connectivity service is required.</li> <li>• <code>internet</code></li> <li>• <code>private-link</code> To check the availability of the <code>private-link</code> proxy type, see <a href="#">SAP Private Link Service</a> in the SAP Discovery Center.</li> </ul>

The following guidelines apply to the configuration of the additional properties of a destination:

## Additional Properties of a Destination

Property	Description
HTML5.ForwardAuthToken	If <code>true</code> , the OAuth token is sent to the destination. The default value is <code>false</code> . This token contains the user identity, scopes, and other attributes. It's signed by the UAA so it can be used for user authentication and authorization with back-end services.
	<p><b> ⓘ Note</b></p> <p>If the <code>ProxyType</code> is set to <code>on-premise</code>, don't set the <code>ForwardAuthToken</code> property to <code>true</code>.</p> <p>If the <code>Authentication</code> is other than <code>NoAuthentication</code>, don't set the <code>ForwardAuthToken</code> property to <code>true</code>.</p>
HTML5.Timeout	Positive integer representing the maximum time to wait for a response (in milliseconds) from the destination. The default value is 30000 ms.
	<p><b> ⓘ Note</b></p> <p>The timeout value specified also applies to the destination's log out path (if defined), which belongs to the <code>destination</code> property.</p>
HTML5.PreserveHostHeader	If <code>true</code> , the application router preserves the host header in the back-end request. This is expected by some back-end systems like AS ABAP, which don't process <code>x-forwarded-*</code> headers.
HTML5.DynamicDestination	If <code>true</code> , the application router allows this destination to be used dynamically on the host or path level.
HTML5.SetXForwardedHeaders	If <code>true</code> , the application router adds X-Forwarded-(Host, Path, Proto) headers to the back-end request. The default value is <code>true</code> .
sap-client	If <code>true</code> , the application router propagates the <code>sap-client</code> and its value as a header in the back-end request.  This is expected by ABAP back-end systems.
HTML5.IASDependencyName	Configures the name of the Identity Authentication dependency that is used to exchange the Identity Authentication token used for the user authentication at login. If configured, the exchanged token is then also forwarded to the backend application.
HTML5.ForwardAuthCertificates	If <code>true</code> , the certificates and key of the authentication service are added to the HTTP connection to the destination. The default value is <code>false</code> . For more information see: <a href="#">Mutual TLS Authentication (mTLS) and Certificates Handling [page 478]</a> .

- If a destination with the same name is defined both in the environment destination and the destination service, the destination configuration loads the settings from the environment.

- If the configuration of a destination is updated in runtime, the changes are reflected automatically to the AppRouter. There's no need to restart the AppRouter.
- The destination service is only available in the Cloud Foundry environment.
- You can define destinations at the service instance level. This destination type has a higher priority than the same destination defined on the subaccount level. This enables exposing a specific destination to a specific application instead of to the entire subaccount.

## Connectivity

Application Router supports integration with the SAP Connectivity service. The Connectivity service handles proxy access to the Cloud Connector, which tunnels connections to private network systems. In order to use connectivity, a connectivity service instance must be created and bound to the Application Router application. In addition, the relevant destination configurations must contain `proxyType=OnPremise` and a valid XSUAA login token must be obtained from the login flow.

## Configure the `IASDependencyName` Property to Enable App-to-App Navigation

You can enable an application to consume the APIs of another application. For more information, see: [Configure Integration Between Applications](#) and [Consume APIs from Other Applications](#)

To enable the technical communication between the applications, the application router must exchange the Identity Authentication token that was used for the user authentication at login to the application with the Identity Authentication token used for the other application. The exchanged token is then forwarded by the application router to the backend applications.

For this, you must configure either a new destination with an `HTML5.IASDependencyName` property that points to the backend application or a destination environment variable with the `IASDependencyName` property (see [Environment Variables \[page 463\]](#)). When the application router has to send a request to the backend destination, the application router uses the `IASDependencyName` property to trigger the exchange of the token and the exchanged token is then forwarded to the backend application in the `Authorization` header.

## Related Information

[Application Router \[page 409\]](#)

[Resource Files \[page 413\]](#)

[Consuming the Destination Service](#)

[Consuming the Connectivity Service](#)

## 4.1.4.2.2.7 User API Service

The application router exposes a user API that returns the details of the users who are logged in to the application.

You implement the user API by modelling an [xs-app.json route \[page 425\]](#).

The user API supports two endpoints:

- `/currentUser` returns all details of logged in users.
- `/attributes` returns the main user properties.

The `/currentUser` endpoint response has the following format:

### ↔ Sample Code

```
```
{
  "firstname": "John",
  "lastname": "Doe",
  "email": "john.doe@sap.com",
  "name": "john.doe@sap.com",
  "displayName": "John Doe (john.doe@sap.com)" (The user ID in the identity provider),
  "scopes": ["openid", "user_attributes", "uaa.user"] (Only if the authentication type is "xsuaa")
}
````
```

The `/attributes` endpoint response has the following format:

### ↔ Sample Code

```
```
{
  "firstname": "John",
  "lastname": "Doe",
  "email": "john.doe@sap.com",
  "name": "john.doe@sap.com" (The user ID in the identity provider),
  "scopes": ["openid", "user_attributes", "uaa.user"] (Only if the authentication type is "xsuaa"),
  < user attributes including custom attributes > (Only if the authentication type is "xsuaa")
}
````
```

### ⓘ Note

The `"name"` property is the user ID in the identity provider, which in many cases is also the email address.

### ⓘ Note

If you specify `"xsuaa"` as the authentication type for the route, the following applies:

- User scopes from the xsuaa access token are added to the response of both endpoints (`/currentUser` and `/attributes`).
- User attributes from the identity provider (IdP) chosen for the authentication are added to the response of the `/attributes` endpoint. If a custom IdP is configured for SAP Cloud Identity

Services – Identity Authentication, the custom user attributes are also added to the response of the `/attributes` endpoint. For more information about the definition of user attributes in Identity Authentication, see [.User Attributes](#).

- To get the user attributes from the custom IdP, add the following property to `xs-security.json` file of the application router: `"foreign-scope-references": ["user_attributes"]`

## Configuring the User API Service in the Routing Configuration File

You implement the user API by modelling an `xs-app.json` route [\[page 425\]](#) using the `sap-approuter-userapi` service.

The following example handles both endpoints:

### ↔ Sample Code

```
{  
  "source": "^/user-api(.*)",  
  "target": "$1",  
  "service": "sap-approuter-userapi"  
}
```

The following example uses only the `/currentUser` endpoint:

### ↔ Sample Code

```
{  
  "source": "^/user-api/currentUser$",  
  "target": "/currentUser",  
  "service": "sap-approuter-userapi"  
}
```

## Related Information

[routes \[page 425\]](#)

## 4.1.4.2.2.8 Integration with HTML5 Application Repository

The application router is integrated with the HTML5 Application Repository service, to retrieve all the static content and routes (`xs-app.json`) of the HTML5 applications stored in the repository.

To integrate HTML5 Application Repository with an application router, create an instance of the `html5-apps-repo` service of the `app-runtime` plan, and bind it to the application router.

Model the `xs-app.json` routes that are used to retrieve static content from HTML5 Application Repository in the following way:

#### ↔ Sample Code

```
{  
    "source": "^(/.*)",  
    "target": "$1",  
    "service": "html5-apps-repo-rt",  
    "authenticationType": "xsuaa"  
}
```

In case the application router needs to serve HTML5 applications not stored in HTML5 Application Repository, it is possible to model that in the `xs-app.json` file of the application router.

When the application router is bound to HTML5 Application Repository, the following restrictions apply:

- It is not possible to implement the "first" middleware slot to provide routes dynamically.
- Only the `workingDir` option can be provided in the start of the application router.
- A mixed scenario of modeling part of the static content in a local resources folder and also retrieving static content from HTML5 Application Repository is not supported.

## Related Information

### 4.1.4.2.2.8.1 Processing a Request at Runtime

At runtime, the application router tries to fetch the `xs-app.json` file from the HTML5 application in the HTML5 Application Repository, and to use it for routing the request.

A valid request to an application router that uses the HTML5 Application Repository must have the following format:

#### ↔ Sample Code

```
https://<tenantId>.<appRouterHost>.<domain>/<bsPrefix>.<appName-appVersion>/  
<resourcePath>
```

For example:

#### ↔ Sample Code

```
https://tenant1.myapprouter.cf.sap.hana.com/comsapappcountry.countrylist/  
index.html
```

| Placeholder   | Mandatory | Description  |
|---|-----------|--|
| bsPrefix  | No        | Used when the application is provided by a business service bound to this approuter.   |
| appName   | Yes       | Used to uniquely identify the application in HTML5 Application Repository.   |
| <p><b> ⓘ Note</b></p> <p>Must not contain dots or special characters.</p> |           |  |
| appVersion  | No        | Used to uniquely identify a specific application version in HTML5 Application Repository. If no version is provided, the default application version will be used. |
| resourcePath  | Yes       | The path to the file as it was stored in HTML5 Application Repository.   |

When processing a request at runtime, the following algorithm is applied:

- If no HTML5 application is found in HTML5 Application Repository for the current request, the central application router `xs-app.json` will be used for routing.
- If the HTML5 application exists in HTML5 Application Repository but no `xs-app.json` file is returned, an error message will be issued and the request processing will be stopped.

## 4.1.4.2.2.8.2 Multitenancy of HTML5 Application Repository

The HTML5 Application Repository is a multitenant service. Non-public HTML5 Applications are visible only to the application providers (with provider subaccounts) and the consumers subscribed to the applications (with consumer subaccounts).

When a multitenant application router is subscribed to a subaccount, the HTML5 Application Repository app-runtime instance that is bound to the application router is returned as a dependency, which triggers the subscription to the app-runtime instance. You can also bind HTML5 Application Repository app-host service instances to the application router to enable the subscription of the corresponding HTML5 applications. During runtime, the application router creates an HTML5 Application Repository app-runtime `client_credentials` token using the tenant URL that the application router determines from the `<TENANT_HOST_PATTERN>` environment variable.

**ⓘ Note**

The creation of the token can fail if the app-runtime instance is not subscribed to the subaccount, which happens if, for example, the application router was subscribed to the subaccount before the HTML5 Application Repository became a multitenant service. In this case the application router will create the token using the provider subaccount.

You can trigger the subscription to the HTML5 Application Repository app-runtime instance by using the SAAS Provisioning API: <https://api.sap.com/api/APISaasManagerService/resource>.

```
PATCH /saas-manager/v1/application/tenants/{tenantId}/subscriptions
```

### Note

If you have an old application router version, the HTML5 Application Repository app-runtime `client_credentials` token is created by using HTML5 application provider subaccount.

## Related Information

[Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#)

### 4.1.4.2.2.9 Integration with Business Services

Application router supports integration with Business Services, which are a flavor of reuse-services.

An SAP Business Service exposes its binding information in a set of attributes in the VCAP\_SERVICES credentials block that enable application router to serve Business Service UI and/or data.

To access business services, the following applies:

- The Business Service UI must be stored in HTML5 Application Repository to be accessible from an application router.
- The Business Service UI must be defined as "public" to be accessible from an application router in a different space than the one from which the UI was uploaded.
- The Business Service data can be served using two grant types:

| Grant Type                      | Description   |
|---------------------------------|---|
| <code>user_token</code>         | The application router performs a token exchange between the login JWT token and the Business Service token, and uses it to trigger a request to the Business Service endpoint. |
| <code>client_credentials</code> | The application router generates a <code>client_credentials</code> token and uses it to trigger a request to the Business Service endpoint.                                     |

To bind a Business Service instance to the application router, provide the following information in the VCAP\_SERVICES credentials:

| Information                          | Mandatory | Description   |
|--------------------------------------|-----------|---|
| <code>sap.cloud.service</code>       | Yes       | Service name as referenced from <code>xs-app.json</code> route and business service prefix, if provided by the Business Service UI.             |
| <code>sap.cloud.service.alias</code> | No        | Short service name alias for user friendly URL business service prefix. Make sure the alias is unique in the context of the application router. |
| <code>endpoints</code>               | No        | One or more endpoints that can be used to access Business Service data.   |

| Information                            | Mandatory | Description   |
|--|-----------|---|
| html5-apps-repo                        | No        | The <code>html5-apps-repo.app_host_id</code> contains one or more <code>html5-apps-repo</code> service instance GUIDs that can be used to retrieve Business Service UIs.  |
| saasregistryenabled                    | No        | Indicates that this Business Service supports SaaS Registry subscription. If provided, the application router returns this Business Service <code>xsappname</code> in the SaaS Registry <code>getDependencies</code> callback.  |
| grant_type                             | No        | The grant type that should be used to trigger requests to the Business Service. Allowed values: <code>user_token</code> (default) or <code>client_credentials</code> .  |
| forwardias token                       | No        | This flag that indicates if, in addition to the exchanged JWT token created by the SAP Authorization and Trust Management Service (xsuaa), the OIDC access token created by Identity Authentication should be forwarded as well. The Identity Authentication token is forwarded in the request header <code>x-ias-token</code> .  |
| forwardias authentication              | No        | This flag indicates that if the login was performed using Identity Authentication, the application router will not exchange the JWT token created by the SAP Authorization and Trust Management Service (XSUAA). Instead, the OIDC access token created by Identity Authentication will be forwarded in the authorization header. |
| URL.headers.<header-name>:header-value | No        | If you provide this information, the application router propagates this attribute as the header to the business service backend. Existing request headers will not be overwritten.  |

The value of the `endpoints` is an object containing the following properties:

| Property | Type   | Optional | Default | Description   |
|----------|--------|----------|---------|---|
| url      | String |          |         | URL to access the Business Service data.  |
| timeout  | Number | X        | 30000ms | Positive integer representing the maximum wait time for a response (in milliseconds) from the Business Service. |

The following example shows how to provide the required information. This information should be provided via the `onBind` hook in the service-broker implementation:

### ↔ Sample Code

```
"country": [
  {
    ...
  },
  "credentials": {
```

```

    "sap.cloud.service": "com.sap.appbasic.country",
    "sap.cloud.service.alias": "country",
    "endpoints": {
        "countrieservice": { "url": "https://icf-countriesapp-test-
service.cfapps.sap.hana.ondemand.com/odata/v2/countrieservice" },
        "countryconfig": {
            "url": "https://icf-countriesapp-test-
service.cfapps.sap.hana.ondemand.com/rest/v1/countryconfig",
            "timeout": 120000
        }
    },
    "html5-apps-repo": {
        "app_host_id": "1bd7c044-6cf4-4c5a-b904-2d3f44cd5569,
1cd7c044-6cf4-4c5a-b904-2d3f44cd54445"
    },
    "saasregistryenabled": true,
    "grant_type": "user_token"
    ...

```

## Related Information

[Accessing Business Service Data \[page 461\]](#)

[Accessing Business Service UI \[page 462\]](#)

[services \[page 443\]](#)

### 4.1.4.2.2.9.1 Accessing Business Service Data

This section describes how the application router accesses the Business Service data.

To access Business Service data, the `xs-app.json` file should have a route referencing a specific `sap.cloud.service` or `sap.cloud.service.alias` via the `service` attribute. If an `endpoint` attribute is also modeled, it will be used to get the service URL; otherwise the fallback URL or `URI` attribute will be used.

#### Sample Code

```

"routes": [
    {
        "source": "/odata/v2/(.*)$",
        "target": "$1",
        "service": "com.sap.appbasic.country",
        "endpoint": "countrieservice"
    },

```

In order to support JWT token exchange, the login JWT token should contain the `uaa.user` scope. This requires that the `xs-security` configuration file contain a role template that references the `uaa.user` scope.

#### Sample Code

```

{
    "xsappname": "simple-approuter",
    "tenant-mode": "shared",
    "scopes": [

```

```

        {
            "name": "uaa.user",
            "description": "UAA"
        },
        {
            "name": "$XSAPPNAME.simple-approuter.admin",
            "description": "Simple approuter administrator"
        }
    ],
    "role-templates": [
        {
            "name": "Token_Exchange",
            "description": "UAA",
            "scope-references": [
                "uaa.user"
            ]
        },
        {
            "name": "simple-approuter-admin",
            "description": "Simple approuter administrator",
            "scope-references": [
                "$XSAPPNAME.simple-approuter.admin"
            ]
        }
    ]
}

```

## Related Information

[Integration with Business Services \[page 459\]](#)

[Accessing Business Service UI \[page 462\]](#)

### 4.1.4.2.2.9.2 Accessing Business Service UI

This section provides information about accessing Business Services UIs that are stored in HTML5 Application Repository.

Business Service UI's must be stored in HTML5 Application Repository and defined in their `manifest.json` files as `public: true` in order to be accessible from an application router application that is typically running in a different space than the Business Service space. In addition, dataSource URIs must be relative to the base URL, which means there is no need for a slash as the first character.

The following is an example `manifest.json` file of a Business Service:

#### ↔ Sample Code

```
{
  "sap.app": {
    "id": "com.sap.appbasic.country.list",
    "applicationVersion: {
      "version": "1.0.0"
    },
    "dataSources": {
      "mainService": {

```

```

        "uri": "odata/v2/countryservice",
        "type": "OData"
    },
    "sap.cloud": {
        "public": true,
        "service": "com.sap.appbasic.country"
    }
}

```

A Business Service that exposes UI, must provide one or more app-host GUIDs in an `html5-apps-repo` block in VCAP\_SERVICES credentials.

To access the Business Service UI, the URL request to the application router must contain a business service prefix, as in the following example of a request URL:

#### ↔ Sample Code

```
https://tenant1.approuter-repo-examples.cfapps.sap.hana.ondemand.com/
comsapappbasiccountry.comsapappbasicscountrylist/test/flpSandbox.html
```

In this example, `comsapappbasiccountry` is the business service prefix which matches the `sap.cloud.service` attribute in the country service VCAP\_SERVICES credentials (without dots). The `comsapappbasicscountrylist` is the name of the HTML5 application as defined in the `app.id` attribute in the `manifest.json` file (without dots).

## Related Information

[Integration with Business Services \[page 459\]](#)

[Accessing Business Service Data \[page 461\]](#)

## 4.1.4.2.2.10 Environment Variables

A list of environment variables that can be used to configure the application router.

The following table lists the environment variables that you can use to configure the application router. The table also provides a short description of each variable and, where appropriate, an example of the configuration data.

Application Router Configuration Variables

| Variable                  | Description   |
|---------------------------|---|
| <code>httpHeaders</code>  | Configures the application router to return additional HTTP headers in its responses to client requests |
| <code>destinations</code> | Provides information about the available application (microservice) destinations.                       |

| Variable                                      | Description   |
|---|---|
| <a href="#">cookies [page 469]</a>            | <p>The application router generates the following third-party cookies and sends them to the client:</p> <ul style="list-style-type: none"> <li>• <code>locationafterLogin</code></li> <li>• <code>fragmentAfterLogin</code></li> <li>• <code>signature</code></li> <li>• <code>JSESSION ID</code></li> </ul> <p>To control the behavior and usage of the third-party cookies, you can configure the attributes <code>SameSite</code> and <code>Partitioned</code>.</p> <p>For more information about third-party cookies, please see the KBA <a href="#">3409306</a>.</p> |
| <code>SESSION_TIMEOUT</code>                  | Sets the time to trigger an automatic central log out from the User Account and Authentication (UAA) server.  |
| <code>SEND_XFRAMEOPTIONS</code>               | Sets or changes the <code>X-Frame-Options</code> header   |
| <code>CJ_PROTECT_WHITELIST</code>             | A list of allowed server or domain origins to use when checking for click-jacking attacks.  |
| <code>WS_ALLOWED_ORIGINS</code>               | A list of the allowed server (or domain) origins that the application router uses to verify requests.   |
| <code>JWT_REFRESH</code>                      | Configures the automatic refresh of the JSON Web Token (JWT) provided by the User Account and Authentication (UAA) service to prevent expiry (default is 5 minutes).  |
| <code>UAA_SERVICE_NAME</code>                 | Specifies the <b>exact</b> name of the UAA service to bind to an application.   |
| <code>INCOMING_CONNECTION_TIM<br/>EOUT</code> | Specifies the maximum time (in milliseconds) for a client connection. If the specified time is exceeded, the connection is closed.  |
| <code>TENANT_HOST_PATTERN</code>              | Defines a regular expression to use when resolving tenant host names in the request's host name.  |
| <code>COMPRESSION</code>                      | Configures the compression of resources before a response to the client.  |
| <code>SECURE_SESSION_COOKIE</code>            | Configures the enforcement of the <code>Secure</code> flag of the application router's session cookie.  |
| <code>REQUEST_TRACE</code>                    | Enables additional traces of the incoming and outgoing requests.  |
| <code>EXTERNAL_REVERSE_PROXY</code>           | Indicates the use of the application router behind an external reverse proxy outside of the Cloud Foundry domain. For more information, see <b>Customer Header</b> under <a href="#">Headers [page 449]</a> .   |
| <code>CORS</code>                             | Provides support for cross-origin requests, for example, by allowing the modification of the request header.  |

| Variable   | Description   |
|--|---|
| DIRECT_ROUTING_URI_PATT<br>ERNS  | Defines a list of URIs that are directed to the routing configuration file (xs-app.json file) of the application router instead of to a specific application's xs-app.json file that is stored in the HTML5 Application Repository. This configuration improves the application loading time and monitoring options.  |
| EXT_SESSION_MGT  | You can configure external session management. See <a href="#">External Session Management [page 478]</a> .   |
| CF_NODEJS_LOGGING_LEVEL  | Sets the minimal logging level of the cf-nodejs-logging-support library of the application router.  |
| STATE_PARAMETER_SECRET   | Enables the use of state parameters to prevent CRFS attacks.<br><br>If this environment variable is set, the application router creates a state parameter for each initial authorization request. By validating that the authentication server returns the same state parameter in its response, the application server can verify that the response did not originate from a third party.  |
| HTTP2_SUPPORT  | Enables the application router to start as an HTTP/2 server.  |
| <p><b>① Note</b></p> <p>To configure HTTP/2 support, you must use Cloud Foundry routes with an HTTP/2 destination protocol. See <a href="#">Configuring HTTP/2 Support</a> in the Cloud Foundry Documentation.</p> <p>As connection-specific header fields aren't supported by the HTTP/2 protocol (see <a href="https://datatracker.ietf.org/doc/html/rfc9113">https://datatracker.ietf.org/doc/html/rfc9113</a>), the application router removes such headers automatically when they are returned from a backend to prevent a failure of the HTTP/2 response.</p> |   |
| FULL_CERTIFICATE_CHAIN   | Enables the application router to send the entire chain of certificates provided in the binding of the authentication service (XSUAA or Identity Authentication) to the backend applications. The certificates are required for mutual TLS authentication (mTLS handshake). See <a href="#">Mutual TLS Authentication (mTLS) and Certificates Handling [page 478]</a> .   |
| OWN_SAP_CLOUD_SERVICE  | An array that contains the business solutions ("SAP cloud services") that your HTML5 applications are associated with as values. This configuration enables the standalone application router to use the same standardized format for runtime URLs (/<sap.cloud.service>.<appId>-<versionId>/) that is also used by the managed application router. If a runtime URL contains one of the defined values in the <sap.cloud.service> section, the application router will recognize the value during the processing of a request. |

| Variable                  | Description  |
|---------------------------|--|
| DYNAMIC_IDENTITY_PROVIDER | <p>Configures a dynamic identity provider.</p> <p>If DYNAMIC_IDENTITY_PROVIDER is set to true, the end user can set the identity provider (IDP) for the application's login process. If SAP Authorization and Trust Management is used, the sap_idprequest query parameter must be filled with the IDP origin key. If Identity Authentication is used, the sap_idp request query parameter must be filled with the name of the corporate identity provider configured in the administration console for SAP Cloud Identity services. Note that this is not the <i>display name</i>, but the <i>name</i> of the corporate identity provider (see <a href="#">Creating URL To Access Application with Specific Identity Provider</a> ).</p> <p>If the identityProvider property is defined in the <a href="#">route [page 425]</a>, its value will be overwritten by the sap_idp query parameter value.</p> <p>The default value for DYNAMIC_IDENTITY_PROVIDER is false.</p> |

## httpHeaders

If configured, the application router sends additional HTTP headers in its responses to a client request. You can set the additional HTTP headers in the `<httpHeaders>` environment variable. The following example configuration shows how to configure the application router to send two additional headers in the responses to the client requests from the application `<myApp>`:

```
cf set-env <myApp> httpHeaders "[ { \"X-Frame-Options\": \"ALLOW-FROM http://acme.com\" }, { \"Test-Additional-Header\": \"1\" } ]"
```

or

```
cf set-env <myApp> httpHeaders '[{ "X-Frame-Options": "ALLOW-FROM http://acme.com" }, { "Test-Additional-Header": "1" }]'
```

### → Tip

To ensure better security of your application set the Content-Security-Policy header. This is a response header which informs browsers (capable of interpreting it) about the trusted sources from which an application expects to load resources. This mechanism allows the client to detect and block malicious scripts injected into the application. A value can be set via the `<httpHeaders>` environment variable in the additional headers configuration. The value represents a security policy which contains directive-value pairs. The value of a directive is an allowlist of trusted sources.

Usage of the Content-Security-Policy header is considered second line of defense. An application should always provide proper input validation and output encoding.

## destinations

The destinations configuration is an array of objects that is defined in the `destinations` environment variable. A destination is required for every application (microservice) that is part of the business application. The following table lists the properties that can be used to describe the destination:

Destination Environment Variable Properties

| Property  | Type    | Mandatory | Description   |
|---|---------|-----------|---|
| <code>name</code>   | String  | Yes       | A unique identifier for the destination   |
| <code>url</code>  | String  | Yes       | The Unique Resource Locator for the application (microservice)  |
| <code>proxyHost</code>  | String  | No        | The host of the proxy server used in case the request should go through a proxy to reach the destination.   |
| <p style="text-align: right;">→ Tip</p> <p>Mandatory if <code>proxyPort</code> is defined.</p>  |         |           |   |
| <code>proxyPort</code>  | String  | No        | The port of the proxy server used in case the request should go through a proxy to reach the destination.   |
| <p style="text-align: right;">→ Tip</p> <p>Mandatory if <code>proxyHost</code> is defined.</p>  |         |           |   |
| <code>forwardAuthToken</code>   | Boolean | No        | If true, the OAuth token will be sent to the destination. The default value is "false". This token contains the user identity, scopes, and some other attributes. The token is signed by the User Account and Authorization (UAA) service so that the token can be used for user-authentication and authorization purposes by backend services. |
| <code>strictSSL</code>  | Boolean | No        | Configures whether the application router should reject untrusted certificates. The default value is "true".  |
| <p style="text-align: right;">⚠ Caution</p> <p>For testing purposes only. Do <b>not</b> use this property in production environments!</p> |         |           |   |

| Property                             | Type    | Mandatory | Description   |
|--------------------------------------|---------|-----------|---|
| timeout                              | Number  | No        | <p>A positive integer representing the maximum amount of time to wait for a response (in milliseconds) from the specified destination. The default is 30000ms.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"> <b>→ Tip</b> <p>The timeout specified here also applies to the logout path, <code>logoutPath</code>, if the logout path is defined, for example, in the application's descriptor file <code>xs-app.json</code>.</p> </div>   |
| proxyType                            | String  | No        | <p>Indicates if the destination is used to access applications in private networks (for example, on-premise) or publicly available on the Internet. Possible value: <code>onPremise</code>. If <code>proxyType</code> is not specified, the default (Internet access) is assumed.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"> <b>→ Tip</b> <p>In Cloud environments, if you set the application's destination <code>proxyType</code> to <code>onPremise</code>, a binding to the SAP BTP connectivity service is required, and the <code>forwardAuthToken</code> property must not be set.</p> </div> |
| <code>IASDependencyName</code>       | String  | No        | <p>Configures the name of the Identity Authentication dependency that is used to exchange the Identity Authentication token used for the user authentication at login. If configured, the exchanged token is then also forwarded to the backend applications.</p>   |
| <code>forwardAuthCertificates</code> | Boolean | No        | <p>If <code>true</code>, the certificates and key of the authentication service are added to the HTTP connection to the destination. The default value is <code>false</code>. For more information see: <a href="#">Mutual TLS Authentication (mTLS) and Certificates Handling [page 478]</a></p>   |

If `true`, the certificates and key of the authentication service are added to the HTTP connection to the destination. The default value is `false`. For more information see: [Mutual TLS Authentication \(mTLS\) and Certificates Handling \[page 478\]](#)

The following example shows a simple configuration for the `<destinations>` environment variable:

#### ↔ Sample Code

```
[  
  {  
    "name" : "ui5",  
    "url" : "https://sapui5.acme.com",  
    "proxyHost" : "proxy",  
    "proxyPort" : "8080",  
    "forwardAuthToken" : false,  
    "timeout" : 1200  
  }  
]
```

It is also possible to include the destinations in the `manifest.yml` file, as illustrated in the following example:

#### ↔ Sample Code

```
- name: node-hello-world  
  memory: 100M  
  path: web  
  env: destinations: >  
    [  
      {"name": "ui5", "url": "https://sapui5.acme.com"}  
    ]
```

## Additional Cookie Attributes

If configured, the application router sends additional cookie attributes in its responses to the client. Additional cookies attributes can be set in the `<COOKIES>` environment variable.

Example of configuration for cookies in the `manifest.yml`:

#### ↔ Sample Code

```
env:  
  COOKIES: >  
    {  
      "SameSite": "None",  
      "Partitioned":  
        {  
          "supportedPartitionAgents": "^(Mozilla.*\\(Chrome|Chromium\\)) /  
          ((109)|(1[1-9][0-9])|([2-9][0-9][0-9]))",  
          "unsupportedPartitionAgents": "PostmanRuntime/7.29.2"  
        }  
    }
```

In this example, the application router sets the `SameSite` attribute of the cookie to `None` and specifies a `Partitioned` attribute that is sent in the responses to the client.

#### ⓘ Note

Currently, only the values `None` and `Lax` are supported for the `SameSite` attribute. The value `Strict` is not supported.

The `Partitioned` attribute contains two required regular expressions:

- `supportedPartitionAgents` for supported agents
- `unsupportedPartitionAgents` for unsupported agents

You can use a wildcard '(.)' in `supportedPartitionAgents` to allow all agents to use the `Partitioned` attribute.

#### ⓘ Note

`unsupportedPartitionAgents` overwrites the configurations in `supportedPartitionAgents`. If an agent is allowed in `supportedPartitionAgents` but disallowed in `unsupportedPartitionAgents`, the `Partitioned` attribute will not be returned.

## SESSION\_TIMEOUT

You can configure the triggering of an automatic central log-out from the User Account and Authentication (UAA) service if an application session is inactive for a specified time. A session is considered to be inactive if no requests are sent to the application router. The following command shows how to set the environment variable `<SESSION_TIMEOUT>` to 40 (forty) minutes for the application `<myApp1>`:

```
cf set-env <myApp1> SESSION_TIMEOUT 40
```

#### ⓘ Note

You can also set the session timeout value in the application's `manifest.yml` file, as illustrated in the following example:

#### ↪ Sample Code

```
- name: myApp1
  memory: 100M
  path: web
  env:
    SESSION_TIMEOUT: 40
```

#### → Tip

If the authentication type for a route is set to "`xsuaa`" (for example, "`authenticationType` : "`xsuaa`")`,` the application router depends on the UAA server for user authentication, and the UAA server might have its own session timeout defined. To avoid problems caused by unexpected timeouts, it is recommended that the session timeout values configured for the application router and the UAA are identical."

## SEND\_XFRAMEOPTIONS

By default, the application router sends the `X-Frame-Options` header with the value "SAMEORIGIN". You can change this behavior either by disabling the sending of the default header value (for example, by setting

SEND\_XFRAMEOPTIONS environment variable to false) or by overriding the value, for example, by configuring additional headers (with the `<httpHeaders>` environment variable).

The following example shows how to disable the sending of the x-Frame-Options for a specific application, `myApp1`:

```
cf set-env <myApp1> SEND_XFRAMEOPTIONS false
```

## CJ\_PROTECT\_WHITELIST

The `<CJ_PROTECT_WHITELIST>` specifies a list of origins (for example, host or domain names) that do not need to be protected against click-jacking attacks. This list of allowed host names and domains are used by the application router's allowlist service to protect XS advanced applications against click-jacking attacks. When an HTML page needs to be rendered in a frame, a check is done by calling the allowlist service to validate if the parent frame is allowed to render the requested content in a frame. The check itself is provided by the allowlist service

The following example shows how to add a host name to the click-jacking protection allowlist for the application, `myApp1`:

```
cf set-env <myApp1> CJ_PROTECT_WHITELIST {<protocol>, <hostname>, <portNr>}
```

The content is a JSON list of objects with the properties listed in the following table:

Allowlist of Host and Domain Names

| Property | Type          | Mandatory | Description  |
|----------|---------------|-----------|--|
| protocol | String        | No        | URI scheme, for example "HTTP".  |
| host     | String        | Yes       | A valid host name, for example, acme . com . hostname, or a domain name defined with an asterisk (*) * . acme . com. |
| port     | String/Number | No        | Port string or number containing a valid port.   |

### ⓘ Note

Matching is done against the properties provided. For example, if only the host name is provided, the allowlist service matches all schemata and protocols.

```
xs set-env <myApp1> CJ_PROTECT_WHITELIST {<*.acme.com>}
```

## WS\_ALLOWED\_ORIGINS

When the application router receives an upgrade request, it verifies that the `origin` header includes the URL of the application router. If this is not the case, then an HTTP response with status 403 is returned to the client. This origin verification can be further configured with the environment variable `<WS_ALLOWED_ORIGINS>`, which defines a list of the allowed origins the application router uses in the verification process.

### Note

The structure of the `<WS_ALLOWED_ORIGINS>` variable is the same as the variable `<CJ_PROTECT_WHITELIST>`.

```
cf set-env <myApp1> WS_ALLOWED_ORIGINS {<*.acme.com>}
```

## JWT\_REFRESH

The `JWT_REFRESH` environment variable is used to configure the application router to refresh a JSON Web Token (JWT) for an application, by default, 5 minutes before the JWT expires, if the session is active.

```
cf set-env <myApp1> JWT_REFRESH 1
```

If the JWT is close to expiration and the session is still active, a JWT refresh will be triggered `<JWT_REFRESH>` minutes before expiration. The default value is 5 minutes. To disable the automatic refresh, set the value of `<JWT_REFRESH>` to 0 (zero).

## UAA\_SERVICE\_NAME

The `UAA_SERVICE_NAME` environment variable enables you to configure an instance of the User Account and Authorization service for a specific application, as illustrated in the following example:

```
cf set-env <myApp1> UAA_SERVICE_NAME <myUAAServiceName>
```

### Note

The details of the service configuration are defined in the `<VCAP_SERVICES>` environment variable, which is not configured by the user.

## INCOMING\_CONNECTION\_TIMEOUT

The `INCOMING_CONNECTION_TIMEOUT` environment variable enables you to set the maximum time (in milliseconds) allowed for a client connection, as illustrated in the following example:

```
cf set-env <myApp1> INCOMING_CONNECTION_TIMEOUT 60000
```

If the specified time is exceeded, the connection is closed. If `INCOMING_CONNECTION_TIMEOUT` is set to zero (0), the connection-timeout feature is disabled. The default value for `INCOMING_CONNECTION_TIMEOUT` is 120000 ms (2 min).

## TENANT\_HOST\_PATTERN

The `TENANT_HOST_PATTERN` environment variable enables you to specify a string containing a regular expression with a capturing group. The requested host is matched against this regular expression. The value of the first capturing group is used as the tenant Id. as illustrated in the following example:

```
cf set-env <myApp1> TENANT_HOST_PATTERN
```

## COMPRESSION

The `COMPRESSION` environment variable enables you to configure the compression of resources before a response to the client, as illustrated in the following example:

```
cf set-env <myApp1> COMPRESSION
```

Here is a complete example of the compression environment variable:

### Sample Code

```
env:  
  COMPRESSION: >  
    {  
      "enabled": true,  
      "minSize": 2048,  
      "compressResponseMixedTypeContent": true  
    }
```

## SECURE\_SESSION\_COOKIE

The `SECURE_SESSION_COOKIE` can be set to `true` or `false`. By default, the `Secure` flag of the session cookie is set depending on the environment the application router runs in. For example, when the application router is running behind a router that is configured to serve HTTPS traffic, then this flag will be present. During local development the flag is not set.

### ⓘ Note

If the `Secure` flag is enforced, the application router will reject requests sent over unencrypted connections.

The following example illustrates how to set the `SECURE_SESSION_COOKIE` environment variable:

```
cf set-env <myApp1> SECURE_SESSION_COOKIE true
```

## REQUEST\_TRACE

You can enable additional traces of the incoming and outgoing requests by setting the environment variable `<REQUEST_TRACE>"true"`. If enabled, basic information is logged for every incoming and outgoing request to the application router.

The following example illustrates how to set the `REQUEST_TRACE` environment variable:

```
cf set-env <myApp1> REQUEST_TRACE true
```

→ Tip

This is in addition to the information generated by the Node.js package `@sap/logging` that is used by the XS advanced application router.

## CORS

The `CORS` environment variable enables you to provide support for cross-origin requests, for example, by allowing the modification of the request header. Cross-origin resource sharing (CORS) permits Web pages from other domains to make HTTP requests to your application domain, where normally such requests would automatically be refused by the Web browser's security policy.

Cross-origin resource sharing (CORS) is a mechanism that allows restricted resources on a Web page to be requested from another domain (protocol and port) outside the domain (protocol and port) from which the first resource was served. The CORS configuration enables you to define details to control access to your application resource from other Web browsers. For example, you can specify where requests can originate from or what is allowed in the request and response headers. The following example illustrates a basic CORS configuration:

```
[  
  {  
    "uriPattern": "^\\route1$",
    "allowedMethods": [
      "GET"
    ],
    "allowedOrigin": [
      {
        "host": "host.acme.com",
        "protocol": "https",
        "port": 345
      }
    ],
    "maxAge": 3600,
    "allowedHeaders": [
      "Authorization",
      "Content-Type"
    ],
    "exposeHeaders": [
      "customHeader1",
      "customHeader2"
    ],
    "allowedCredentials": true
  }
]
```

The CORS configuration includes an array of objects with the following properties, some of which are mandatory:

#### Available Settings for CORS Options

| CORS Property  | Type   | Mandatory | Description  |
|----------------|--------|-----------|--|
| uriPattern     | String | Yes       | A regular expression (RegExp) representing the source routes to which the CORS configuration applies. To ensure that the RegExp matches the complete path, surround it with ^ and \$. For example, "uriPattern": "\^\\route1\\$". Defaults: none   |
| allowedOrigin  | Array  | Yes       | A comma-separated list of objects each of which contains a host name, port and protocol that are allowed by the server, for example: [ { "host": "www.acme.com" } ] or [ { "host": ".acme.com" } ].  |
|                |        |           | <p><b> ⓘ Note</b></p> <p>Matching is case-sensitive. In addition, if no port or protocol is specified, the default is "*" .</p>  |
|                |        |           | <p>The default configuration is: [ { "host": "*" } ], which means that the server allows <b>any</b> origin to access the resource.</p>   |
| allowedMethods | Array  | No        | A comma-separated list of HTTP methods that are allowed by the server, for example, "GET", "POST". If <code>allowMethods</code> is defined but no method is specified, the default "GET", "POST", "HEAD", "OPTIONS" (all) applies.   |
|                |        |           | <p><b>→ Tip</b></p> <p>The specified methods must be upper-case, for example, GET. Matching of the method type is case-sensitive.</p>  |
| allowedHeaders | Array  | No        | A comma-separated list of request headers that are allowed by the server. The default values are as follows: [ "Origin", "Accept", "X-Requested-With", "Content-Type", "Access-Control-Request-Method", "Access-Control-Request-Headers" ].  |
| maxAge         | String | No        | A single value specifying the length of time (in seconds) a preflight request should be cached for. A negative value that prevents CORS filter from adding this response header to the pre-flight response. If <code>maxAge</code> is defined but no value is specified, the default time of "1800" seconds applies. |

| CORS Property      | Type    | Mandatory | Description  |
|--------------------|---------|-----------|--|
| exposeHeaders      | Array   | No        | A comma-separated list of <b>response</b> headers that are allowed to be exposed. If <code>exposeHeaders</code> is defined but no response header is specified for exposure, no default value is supplied. |
| allowedCredentials | Boolean | No        | A Boolean flag that indicates whether the specified resource supports user credentials. The default setting is "true".   |

It is also possible to include the CORS configuration in either the `manifest.yml` or the `manifest-op.yml` file. The code in the following example enables the CORS configuration for any route with a source URI pattern that matches the RegExp "`^/route1$`":

#### ↔ Sample Code

CORS Configuration in the `manifest.yml` File

```
- name: node-hello-world
  memory: 100M
  path: web
  env:
    CORS: >
      [
        {
          "allowedOrigin": [
            {
              "host": "my_host",
              "protocol": "https"
            }
          ],
          "uriPattern": "^/route1$"
        }
      ]

```

## DIRECT\_ROUTING\_URI\_PATTERNS

With the direct routing URI patterns configuration, you can define a list of URIs that are directed to the routing configuration file (`xs-app.json` file) of the application router instead of to a specific application's `xs-app.json` file that is stored in the HTML5 Application Repository. This configuration improves the application loading time and monitoring options because it prevents unnecessary calls to the HTML5 Application Repository.

The configuration is an array of strings or regular expressions. You have to provide only the first segment in the URL, after the approuter host. For example, for the URL `https://<approuter-host>/route1/index.html`, you enter `route1` in the direct routing URI patterns array.

#### ↔ Sample Code

```
env:
  DIRECT_ROUTING_URI_PATTERNS: >
    [ "route1", "^route2$", "route3" ]
```

## CF\_NODEJS\_LOGGING\_LEVEL

With the CF\_NODEJS\_LOGGING\_LEVEL variable, you can set the minimal logging level of the cf-nodejs-logging-support library of the application router. The following levels are available:

- off
- error
- warn
- info
- verbose
- debug
- silly

The default value is `error`.

Here is a sample content for the CF\_NODEJS\_LOGGING\_LEVEL environment variable:

### ↔ Sample Code

```
env:  
  CF_NODEJS_LOGGING_LEVEL: "debug"
```

## Related Information

[Application Router \[page 409\]](#)

[Routing Configuration File \[page 420\]](#)

## 4.1.4.2.2.10.1 Performance Statistics

The application router provides performance statistics in an HTTP response header if the HTTP request contains an HTTP query parameter (URL parameter) `sap-statistics=true` or if the HTTP request contains an HTTP header field `sap-statistics:true`.

If an HTTP request that contains a header field or query parameter with `sap-statistics=true` reaches the application router, the application router forwards an `sap-statistics` header to the corresponding backend. If SAP statistics is implemented for the backend, the backend returns to the application router a response header containing the statistics information from the backend.

The application router returns the following statistics information in an `sap-statistics` response header:

- `total`: The time that has passed between the moment when the request entered into the application router and the moment when the application router started writing the response
- `ext` (in case of destination forwarding): The time spent in the backend

Each backend sub-component can add its own response header with duration measurements when it receives the HTTP header `sap-statistics:true`.

## 4.1.4.2.2.10.2 Mutual TLS Authentication (mTLS) and Certificates Handling

Mutual TLS (mTLS) is a mutual authentication mechanism. The application router supports the use of certificates for the creation of tokens and an mTLS handshake in backend connections.

To enable the mTLS authentication for backend connections, the following prerequisites must be met:

- The authentication service instance that is bound to the application router, Authorization and Trust Management service (XSUAA) or Identity Authentication (IAS), must provide a certificate chain and a private key in its credentials.  
If the private key isn't provided in the credentials, you can also configure the environment variables XSUAA\_PRIVATE\_KEY (Authorization and Trust Management) and IAS\_PRIVATE\_KEY (Identity Authentication) in the application router to provide the private key.
- The destinations for the backend applications must contain either the `HTML5.ForwardAuthCertificates` property in the configuration provided by the destination service (see [Application Routes and Destinations \[page 450\]](#)) or the `forwardAuthCertificates` property in the destinations environment variable (see [Environment Variables \[page 463\]](#)).
- In Cloud Foundry, the client certificate is propagated using the `x-forwarded-client-cert` header. To enable this, the backend URL must contain a `.cert` segment in its domain.

The application router gets the XSUAA or IAS tokens that provide the certificates chain and private key. The application router uses the certificates chain and private key from the credentials of the authentication service instance for the following:

- The application router creates the HTTP connection to backend using the private key and a chain of intermediate and client certificates to enable the mTLS handshake. If the `FULL_CERTIFICATE_CHAIN` environment variable is enabled, the application router sends the entire chain of certificates provided in the binding of the authentication service (XSUAA or IAS) to the backend applications. If this environment variable is not enabled, the application router sends only the root and intermediate certificates from the chain.
- When forwarding a request to business services, the application router uses these certificates also to create a `client_credentials` token or to exchange the login token.

## 4.1.4.2.2.11 External Session Management

The application router supports the backup of user sessions in an external session store. This enables a session recovery if the application router instance that stores a session crashes and another application router instance needs to continue handling the running user session.

To enable this capability the you must bind a service instance of a service that supports a fast persistence store such as Redis. When such a service is bound, the application router backs up the in memory session information into the external persistency store. If, in subsequent requests, the session information is not found in the in memory session store, the application router tries to rebuild the in memory session store session information from external persistency store.

The sessions are stored encrypted and compressed. For capacity planning, you can assume 50 Kb per session storage in fast persistence store.

## Configuration of External Session Management

To use external session management, you have to set the `EXT_SESSION_MGT` environment variable. The variable value is defined in JSON format, providing the following properties:

- `instanceName` (mandatory): the name of the service instance of the storage service
- `storageType` (mandatory): the type of the storage, for example - "redis".  
Note that if no custom storage driver is used, only `redis` is allowed.
- `sessionSecret` (mandatory): Since the application router stores encrypted sessions in a persistence store, a shared secret shall be provided. Please generate a unique string with at least 64 characters.

For example:

### ↔ Sample Code

```
{  
  "instanceName": "approuter-redis",  
  "storageType": "redis",  
  "sessionSecret": "someuniquesessionsecret"  
}
```

### ⓘ Note

Currently, the application router supports only a Redis store.

## Configuration of a Custom Storage Driver

You can use your own driver. In order to do that, the user injects its own implementation of a store.

The class has to implement the following interface:

### ↔ Sample Code

```
interface UserCustomStore {  
  // delete all sessions  
  clear(): Promise<void>;  
  // remove <sessionId> session  
  destroy(sessionId : string): Promise<void>;  
  // retrieve <sessionId> session  
  get(sessionId : string): Promise<object | null>;  
  // number of sessions  
  length(): Promise<number>;  
  // get <sessionId> expiration  
  ttl(sessionId : string): Promise<number>;  
  // set <sessionId> data to <session> with <timeout> expiration  
  set(sessionId: string, session: object, timeout: number): Promise<void>;  
  // check if session <sessionId> exists  
  exists(sessionId: string): boolean;  
  // update existing session <sessionId> expiration to <timeout>  
  resetTimer(sessionId: string, timeout: number);  
}
```

In addition, the file must include a method to get an instance of the store, for example:

#### ↔ Sample Code

```
let store;
module.exports.getStore = () => {
  if (!store) {
    store = new UserCustomStore();
  }
  return store;
};
```

In order for application router to use it, user has to set the `externalStoreFilePath` property in the `EXT_SESSION_MGT` environment variable with the path to the storage. The application router uses this path to require your storage For example:

#### ↔ Sample Code

```
{
  "instanceName": "approuter-redis",
  "storageType": "redis",
  "sessionSecret": "someuniquesessionsecret",
  "externalStoreFilePath": "./src/storage/my-special-storage"
}
```

## Related Information

[Environment Variables \[page 463\]](#)

### 4.1.4.2.2.12 Multitenancy

Each multitenant application has to deploy its own application router, and the application router handles requests of all tenants to the application. The application router is able to determine the tenant identifier out of the URL and then forwards the authentication request to the tenant User Account and Authentication (UAA) service and the related identity zone.

To use a multitenant application router, you must have a shared UAA service and the version of the application router has to be greater than 2.3.1.

The application router must determine the tenant-specific subdomain for the UAA that in turn determines the identity zone, used for authentication. This determination is done by using a regular expression defined in the environment variable `TENANT_HOST_PATTERN`.

`TENANT_HOST_PATTERN` is a string containing a regular expression with a capturing group. The request host is matched against this regular expression. The value of the first capturing group is used as the tenant subdomain.

If you have multiple routes to the same application, for example:

`tenant1.<application domain>` and `tenant2.<application domain>`

The TENANT\_HOST\_PATTERN could be:

```
TENANT_HOST_PATTERN: "^(.*).<application domain>"
```

With this configuration, the application router extracts the tenant subdomain, which is used for authentication against a multitenant UAA.

## Register an Application (SaaS Registry Configuration)

Create a service instance of the SaaS Provisioning service with a configuration JSON file in the Cloud Foundry sub-account space where the multitenant business application is deployed.

The configuration JSON file must use the following format and set these properties:

```
{  
    "xsappname" : "<xsappname>",  
    "appUrls": {  
        "getDependencies" : "<approuter-host>/<getDependenciesPath>/<tenantId>",  
        "onSubscription" : "<approuter-host>/<onSubscriptionPath>/<tenantId>"  
    },  
    "displayName" : "<display_name>",  
    "description" : "<description>",  
    "category" : "<category>"  
}
```

Specify the following parameters:

| Parameters          | Description  |
|---------------------|--|
| xsappname           | The xsappname configured in the security descriptor file used to create the XSUAA instance (see <a href="#">Develop the Multitenant Application [page 363]</a> ).  |
| getDependenciesPath | (Optional) Any URL that the application exposes for GET dependencies. If the application doesn't have dependencies and the callback isn't implemented, it mustn't be declared.   |
| onSubscriptionPath  | This parameter must end with / {tenantId}. The tenant for the subscription is passed to this callback as a path parameter. You must keep {tenantId} as a parameter in the URL so that it's replaced at runtime with the tenant calling the subscription. This callback URL is called when a subscription between a multitenant application and a consumer tenant is created (PUT) and when the subscription is removed (DELETE). |
| displayName         | (Optional) The display name of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's technical name.  |
| description         | (Optional) The description of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's display name.   |
| category            | (Optional) The category to which the application is grouped in the <i>Subscriptions</i> page in the cockpit. If this parameter is left empty, it's assigned to the default category.   |

### ⓘ Note

For information about how to model the security settings for activating the `saas-registry` callbacks, see [Develop the Multitenant Application \[page 363\]](#).

## Related Information

[Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#)

[Register the Multitenant Application to the SAP SaaS Provisioning Service \[page 370\]](#)

### 4.1.4.2.2.13 Extending the Application Router

Configure application-specific extensions for the application router.

Instead of starting the application router directly, you can configure your XS advanced application to use its own start script. You can also use the application router as a regular Node.js package.

#### ↗ Sample Code

```
var approuter = require('@sap/approuter');
var ar = approuter();
ar.start();
```

## Custom Middleware Injection

The application router uses the connect framework, for more information, see [Connect framework](#) in the *Related Links* below. You can reuse all injected “connect” middleware within the application router, for example, directly in the application start script:

#### ↗ Sample Code

```
var approuter = require('@sap/approuter');
var ar = approuter();
ar.beforeRequestHandler.use('/my-ext', function myMiddleware(req, res, next) {
    res.end('Request handled by my extension!');
});
ar.start();
```

#### → Tip

To facilitate troubleshooting, always provide a name for your custom middleware.

The `path` argument is optional. You can also chain `use` calls.

## ↔ Sample Code

```
var approuter = require('@sap/approuter');
var morgan = require('morgan');
var ar = approuter();
ar.beforeRequestHandler
  .use(morgan('combined'))
  .use('/my-ext', function myMiddleware(req, res, next) {
    res.end('Request handled by my extension!');
  });
ar.start();
```

The application router defines the following slots where you can insert custom middleware:

- `first` - right after the connect application is created, and before any application router middleware. At this point security checks are not performed yet.

### → Tip

This is good place for infrastructure logic, for example, logging and monitoring.

- `beforeRequestHandler` - before standard application router request handling, that is static resource serving or forwarding to destinations.

### → Tip

This is a good place to handle custom REST API requests.

- `beforeErrorHandler` - before standard application router error handling.

### → Tip

This is a good place to capture or customize error handling.

If your middleware does not complete the request processing, call `next` to return control to the application router middleware, as illustrated in the following example:

## ↔ Sample Code

```
ar.beforeRequestHandler.use('/my-ext', function myMiddleware(req, res, next) {
  res.setHeader('x-my-ext', 'passed');
  next();
});
```

## Application Router Extensions

An extension is defined by an object with the following properties:

- `insertMiddleware` - describes the middleware provided by this extension
  - `first`, `beforeRequestHandler`, and `beforeErrorHandler`  
An array of middleware, where each one can be either of the following elements:
    - A middleware function (invoked on all requests)

- An object with the following properties:
  - `path`  
Handle requests only for this path
  - `handler`  
The middleware function to invoke

#### ↳ Sample Code

Example Approuter Extension Configuration (`my-ext.js`)

```
module.exports = {
  insertMiddleware: {
    first: [
      function logRequest(req, res, next) {
        console.log('Got request %s %s', req.method, req.url);
      }
    ],
    beforeRequestHandler: [
      {
        path: '/my-ext',
        handler: function myMiddleware(req, res, next) {
          res.end('Request handled by my extension!');
        }
      }
    ]
  }
};
```

The extension configuration can be referenced in the corresponding application's start script, as illustrated in the following example:

#### ↳ Sample Code

```
var approuter = require('@sap/approuter');
var ar = approuter();
ar.start({
  extensions: [
    require('./my-ext.js')
  ]
});
```

## Customize Command Line

By default the application router handles its command-line parameters, but that can be customized as well.

An `<approuter>` instance provides the property `cmdParser` that is a commander instance. It is configured with the standard application router command line options. You can add custom options like this:

#### ↳ Sample Code

```
var approuter = require('@sap/approuter');
var ar = approuter();
var params = ar.cmdParser
// add here custom command line options if needed
.option('-d, --dummy', 'A dummy option')
```

```
.parse(process.argv);
console.log('Dummy option:', params.dummy);
```

To disable the handling of command-line options in the application router, reset the `ar.cmdParser` property to "false", as illustrated in the following example:

```
ar.cmdParser = false;
```

## Related Information

[Middleware Connect Framework](#)

[Extension API of the Application Router \[page 485\]](#)

### 4.1.4.2.2.14 Extension API of the Application Router

A detailed list of the features and functions provided by the application router extension API.

The application router extension API enables you to create new instances of the application router, manage the approuter instance, and insert middleware using the Node.js "connect" framework. This section contains detailed information about the following areas:

- Approuter Extension API Reference
- Middleware Slot

## Approuter Extension API Reference

The application router uses the "Connect" framework for the insertion of middleware components. You can reuse all connect middleware within the application router directly.

Approuter Extension API Functions

|                          |  |
|--------------------------|--|
| <code>approuter()</code> | Creates a new instance of the application router   |
| <code>first</code>       | Defines a "middleware slot" (a slot for the insertion of middleware) immediately after the <code>connect</code> application is created, and before any application router middleware |

#### → Tip

This is a good place to insert infrastructure logic, for example, logging and monitoring.

|                                       |  |
|---------------------------------------|--|
| <code>beforeRequestHandler</code>     | Defines a “middleware slot” before the standard application router request handling, that is; static resource serving or forwarding to destinations.   |
|                                       | <p><b>→ Tip</b></p> <p>This is a good place to handle custom REST API requests.</p>  |
| <code>beforeErrorHandler</code>       | Defines a “middleware slot” before the standard application router error handling  |
|                                       | <p><b>→ Tip</b></p> <p>This is a good place to capture or customize error handling.</p>  |
| <code>start(options, callback)</code> | <p>Starts the application router with the given options.</p> <ul style="list-style-type: none"> <li>• <code>options</code> this argument is optional. If provided, it should be an object which can have any of the following properties: <ul style="list-style-type: none"> <li>• <code>port</code> - a TCP port the application router will listen to (string, optional)</li> <li>• <code>workingDir</code> - the working directory for the application router, should contain the <code>xs-app.json</code> file (string, optional)</li> <li>• <code>extensions</code> - an array of extensions, each one is an object as defined in Application Router Extensions (optional)</li> <li>• <code>xsappConfig</code> - An object representing the content which is usually put in <code>xs-app.json</code> file. If this property is present it will take precedence over the content of <code>xs-app.json</code>.</li> </ul> </li> <li>• <code>callback</code> - optional function with signature <code>callback(err)</code>. It is invoked when the application router has started or an error has occurred. If not provided and an error occurs (for example the port is busy), the application will abort.</li> </ul> |
| <code>close(callback)</code>          | <p>Stops the application router.</p> <ul style="list-style-type: none"> <li>• <code>callback</code> - optional function with signature <code>callback(err)</code>. It is invoked when the application router has stopped or an error has occurred.</li> </ul>  |

## Middleware Slot

Manage the insertion of middleware slots with the application router.

|                    |  |
|--------------------|--|
| use(path, handler) | Inserts a request handling middleware in the current slot. <ul style="list-style-type: none"><li>• <code>path</code> - handle only requests starting with this path (string, optional)</li><li>• <code>handler</code> - a middleware function to invoke (function, mandatory)</li></ul> Returns this for chaining. |
|--------------------|--|

---

## Related Information

[Middleware Connect Framework](#) ↗

[Extending the Application Router \[page 482\]](#)

### 4.1.4.2.3 Managed Application Router

The managed application router enables you to access and run HTML5 applications in a cloud environment without the need to maintain your own runtime infrastructure.

The managed application router is the HTML5 applications runtime capability that is provided by the following products:

- SAP Build Work Zone, standard edition
- SAP Build Work Zone, advanced edition
- SAP Cloud Portal

To use the managed application router, you must be subscribed to one of these services.

For information how to develop HTML5 applications and run them using the managed application router, see the following product-specific guides:

- [Developing HTML5 Applications](#) for **SAP Build Work Zone, standard edition**
- [Developing HTML5 Applications](#) for **SAP Build Work Zone, advanced edition**
- [Developing HTML5 Applications](#) for **SAP Cloud Portal**

### 4.1.4.2.4 Sizing Guide

This guide explains how much application runtime you must purchase to run HTML5 applications.

To develop and run HTML5 Applications using a standalone application router on Cloud Foundry, customers must purchase the Cloud Foundry application runtime.

The memory consumption for an idle application router is around 50 MB. An application router process should run with at least 256MB memory. It may require more memory depending on the application. These aspects influence the memory usage:

- Number of concurrent users. The number of users that log on to the applications at the same time has the greatest impact on the required runtime.
- Number of active sessions
- JWT token size
- Backend session cookies

Using the following table, estimate how much application runtime allocation you need, based on t-shirt sizes and the expected number of concurrent users.

| T-Shirt Size | Concurrent Users | Required Application Runtime (in GB) |
|--------------|------------------|--------------------------------------|
| S            | 1000             | 1                                    |
| M            | 10000            | 4                                    |
| L            | 30000            | 8                                    |
| XL           | 80000            | 16                                   |

## Testing the Required Memory

The following tests provide measurements for different application router scenarios. You can use the measurements to approximately calculate the amount of memory that will be required by the application router. The tables contain the exact results from the measurements with Node.js v6.9.1. We recommend assuming higher numbers for productive use.

All measurements were done with authentication. If you have additional session content and want to count the session memory consumption, take a look at what is stored in the session (see "Sessions" in [Application Router Configuration \[page 414\]](#)). You must add the calculated session size taking into account the number of different users and the session timeout. In our tests, the JWT token alone required approximately 4 KB.

## Testing the HTTP Traffic

We tested two different test scenarios

- Scenario 1: A 'Hello World' static resource is served.
- Scenario 2:
  - A 'Hello World' static resource is served.
  - A static resource of 84.78 KB (compressed by application router to 28.36 KB) is served.
  - A backend application which returns a payload of 80 kb (compressed by application router to 58 KB) is called.
  - Another backend which returns a payload of 160 KB (compressed by application router to 116 KB) is called.

Results for HTTP Traffic Tests

| Memory Limit | Max. Number of Sessions - Scenario 1 | Max. Number of Sessions - Scenario 2 |
|--------------|--------------------------------------|--------------------------------------|
| 256 MB       | 5300                                 | 800                                  |

| Memory Limit | Max. Number of Sessions - Scenario 1 | Max. Number of Sessions - Scenario 2 |
|--------------|--------------------------------------|--------------------------------------|
| 512 MB       | 13300                                | 2300                                 |
| 1 GB         | 30100                                | 8400                                 |
| 2 GB         | 65500                                | 19500                                |
| 4 GB         | 134900                               | 46400                                |
| 8 GB         | 275500                               | 102300                               |

## Testing Web Socket Traffic

We tested two different test scenarios:

- Scenario 1:
  - A 'Hello World' static resource is served.
  - A single 'Hello' message is sent and then received through a web socket connection.
- Scenario 2:
  - A 'Hello World' static resource is served.
  - A static resource of 84.78 KB (compressed by the application router to 28.36 KB) is served.
  - A backend which returns a payload of 80 KB over a web socket is called.
  - Another backend which returns a payload of 160 KB over a web socket is called.

### ⓘ Note

Web sockets require a certain number of file handles to be available for the process. This number is approximately two times the number of the sessions. In Cloud Foundry, the default value is 16384.

Results for Web Socket Traffic Tests

| Memory Limit | Max. Number of Sessions - Scenario 1 | Max. Number of Sessions - Scenario 1 |
|--------------|--------------------------------------|--------------------------------------|
| 256 MB       | 600                                  | 300                                  |
| 512 MB       | 1100                                 | 500                                  |
| 1 GB         | 3100                                 | 800                                  |
| 2 GB         | 6500                                 | 1400                                 |
| 4 GB         | 13300                                | 2900                                 |
| 8 GB         | 20700                                | 6100                                 |

### ⓘ Note

--max-old-space-size restricts the amount of memory used in the JavaScript heap. Its default value is below 2 GB. To use the full resources that has been provided to the application, the value of this restriction should be set to a number equal to the memory limit of the whole application.

For example, if the application memory is limited to 2 GB, set the V8 heap limit in the package.json file as follows:

```
"scripts": {
  "start": "node --max-old-space-size=2048 node_modules/@sap/approuter/approuter.js"
}
```

## 4.1.4.2.5 Getting Support

If you have questions or encounter an issue while working with HTML5 Application Repository, there are various ways to address them.

Create an incident in the SAP Support Portal using one of the following components:

| Service                      | Component      |
|------------------------------|----------------|
| HTML5 Application Repository | BC-CP-CF-HTML5 |
| Application Router           | BC-CP-APR      |

Provide the following incident details:

### HTML5 Application Repository

Enter and attach the following to the incident;

- Subaccount HTML5 Application Repository/app-host entitlement
- Requested app-host size limit
- Upload your mtad.yaml file
- Upload your html5-app-deployer application logs (cf logs <html5-app-deployerAppName> --recent)

### Application Router

Enter and attach the following to the incident;

- Version
- Working with or without HTML5 Apps Repo
- Upload your Application Router xs-app.json file
- Upload your Application Router xs-security.json file or configuration
- Application Router environment (cf env <approuterApp> or from cockpit).

#### ⓘ Note

The credentials(clientid/clientsecret) should be removed.

- Upload your Application Router logs

## Related Information

[Getting Support for SAP BTP \[page 3148\]](#)  
[Troubleshooting \[page 491\]](#)

### 4.1.4.2.6 Troubleshooting

A troubleshooting guide for HTML5 application repository.

## Uploading Applications

### 400: Application metadata already exists when uploading HTML5 applications

| Term     | Description   |
|----------|---|
| Issue    | HTML5 Application Deployment fails with error "Application metadata for application xyz already exists."  |
| Cause    | There is already an app-host service instance that contains a manifest.json with app.id = xyz in this space it is not possible to have multiple app-hosts containing the same app.id. |
| Solution | Delete or do not deploy the old app-host instance or use another app.id in the resources folder for a new app-host.   |

### 400: Failed to run the application router after subscription; route not found

| Term     | Description  |
|----------|--|
| Issue    | After subscribing to the application router using the <subdomain>-<myapprouter>.scp domain format, the calling the application router fails with error "route not found".  |
| Cause    | In order to support subscriptions out of the box, a route with format *.<custom-domain> should be created and mapped to the application router. The * host represents a wildcard that is replaced by the actual subscriber subdomain during runtime.<br><br>Without a custom domain, this type of route (wildcard host) cannot be created; only a fixed host can be used. Without a custom domain, the route format is <subdomain>-<myapprouter>.scp domain. It means that you have to create a route with the expected subdomain for each new subscriber. |
| Solution | In development setups, if you don't have a custom domain before subscription, then create a route with the expected subscriber subdomain.  |

## **400: Uploading application content failed**

| Term     | Description   |
|----------|---|
| Issue    | When trying to upload content, it fails with error: "Upload failed".  |
| Cause    | <p>One or more of the input validations performed by HTML5 application repository failed. The possible validation failures are:</p> <ol style="list-style-type: none"><li>1. Missing manifest.json file on the root level.</li><li>2. manifest.json app.id has invalid characters (hyphens, @, %, &amp;, etc.).</li><li>3. manifest.json app.version is not using the following format: <code>xx.xx.xx</code>, where x must be an integer (e.g.: -snapshot is not supported).</li><li>4. app.id of one or more of the applications already exists in another html5-apps-repo/app-host service instance in the same space.</li></ol> |
| Solution | Check if one of the causes is your issue. For example, check the size of your html5-app-deployer resources folder, check manifest.json. If you are not sure if another service instance already uses your app.id, try making a small change to your app.id and deploy it again.   |

## **403: App-host is being modified by another process**

| Term     | Description  |
|----------|--|
| Issue    | HTML5 application repository deployment fails with error "app-host is being modified by another process."                                      |
| Cause    | The HTML5 application repository deployer attempts to upload content while another deployer is also uploading content using the same app-host. |
| Solution | Try again after the first HTML5 application repository deployer has completed its upload.  |

## **409: App-host deploy or redeploy remains in progress**

| Term  | Description   |
|-------|---|
| Issue | HTML5 application repository deployment fails with error "Deploy in progress" or "Redeploy in progress" and response type 409 for a long time or any other issue. |
| Cause | If something happens during the upload, it might cause some inconsistencies.  |

| Term     | Description   |
|----------|---|
| Solution | <p>Delete the content of one or more multiple app-hosts, and reset the state to initial without deleting the service instances. Use the following command line in the CF CLI HTML5 application repository plug-in:</p> <pre>cf html5-delete --content &lt;app-host-id&gt; [...]</pre> |

### Failed to upload content; exceeded maximum file length

| Term     | Description   |
|----------|---|
| Issue    | When trying to upload content, it fails with error: "Error while parsing request; Error: maximum file length exceeded".                     |
| Cause    | The zipped application file length exceeds the size limit of the app-host service instance.   |
| Solution | Increase the app-host size limit using the update service. For example: <code>cf update-service my-app-host -c '{"sizeLimit":100}'</code> . |

### Timeout while uploading content

| Term     | Description   |
|----------|---|
| Issue    | When trying to upload content using the <code>cf push</code> command, it fails with error: "Failed to make TCP connection to port 8080: connection refused. Timed out after 1m0s: health check never passed." |
| Cause    | The <code>cf push</code> plugin checks if the start process is finished. If the process times out, then it tries to start it again.   |
| Solution | Set <code>health-check-type</code> to <code>none</code> in the <code>manifest.yaml</code> of the HTML5 Application Deployer.  |

### Uploading application content failed; application size exceeds the maximum size limit of 100 MB

| Term     | Description  |
|----------|--|
| Issue    | When trying to upload content, you receive the following error: "Uploading application content failed: application's size exceeds the maximum size limit of 100 MB". |
| Cause    | The unzipped applications content exceeds the size limit (deprecated) of the app-host service instance.  |
| Solution | Deploy an application that is less than 100 MB or remove some of your applications and move them to another app-host..   |

## Running Applications

### 400: Failed to retrieve xs-app.json; invalid app-host ID

| Term     | Description  |
|----------|--|
| Issue    | Serving content from the HTML5 application repository fails with error "Invalid App Host ID. Please check with business service provider if the requested App Host ID is valid". |
| Caution  | HTML5 application repository belongs to a business service and the app host ID is invalid or incompatible.   |
| Solution | Ask the business service owner to define "public" : true in the app manifest.json or wait until the HTML5 application repository is restarted.                                   |

### 403: Failed to retrieve xs-app.json; unauthorized

| Term     | Description   |
|----------|---|
| Issue    | Serving content from the application router fails with error "Unauthorized. Please check with the business service UI provider if the requested UI is defined as public". |
| Cause    | HTML5 application repository belongs to a business service and it is not public.  |
| Solution | Ask the business service owner to define "public" : true in the app manifest.json.  |

### 404: Application does not exist

| Term  | Description  |
|-------|--|
| Issue | The HTML5 application repository fails to serve content. The application log states: "Application xyz does not exist" is printed to the console.   |
| Cause | <p>The application name provided in URL is not correct or the request URL doesn't provide the application key:</p> <p>Application names are stored in HTML5 application repository without using full stops as separators in the URL. If <i>manifest.json app.id</i> equals <i>country.list</i>, then the application name is <i>countrylist</i> and the same application name should be used in URL.</p> <p>A request to the application router must provide the application key in the URL because the application router uses the application key to fetch the xs-app.json file of the HTML5 application from the HTML5 Application Repository. The application key can consist of the business service prefix, application name, and application version. (Only the application name is mandatory.).</p> |

| Term     | Description   |
|----------|---|
| Solution | <p>First check the application name. For the request URL, use the application name without full stops as separators.</p> <p>If the application name is correct, check if the request URL to the application router contains the application key. If it doesn't contain the application key, check how you configured the backend application data retrieval in your HTML5 application:</p> <ul style="list-style-type: none"> <li>• If you use SAP Fiori tools such as BAS, in the <code>manifest.json</code> file of the application, check the <code>dataSources.uri</code> property. The value for <code>dataSources.uri</code> must not start with a slash ("/"). For example, the value <code>northwind/V2/ Northwind.svc</code> this is correct, but <code>/northwind/V2/ Northwind.svc/</code> is wrong because it would create an absolute path from which the browser cannot concatenate the application key for the request. If there is a slash, remove it.</li> <li>• If you use a JQueryAjax call for the request, make sure that the URL that is provided to the JQueryAjax call is a relative path and does not start with a slash (" / ").</li> </ul> |

#### 404: Calls to service endpoints specified in an application's xs-app.json fails

| Term     | Description  |
|----------|--|
| Issue    | You have defined routes in a UI app's local <code>xs-app.json</code> but calls do not get routed and instead return a 404 error.   |
| Cause    | The routes in the <code>xs-app.json</code> file are processed top to bottom. If a route maps the pattern, it is picked even if a route below it is a better match. The route for the HTML5 application repository typically is a "catch all" route, and if any routes are defined below it, then they are never reached. |
| Solution | Move the route leading to the <code>html5-apps-repo-rt</code> to be the last entry in the <code>xs-app.json</code> file.   |

#### 500: Failed to retrieve xs-app.json

| Term     | Description  |
|----------|--|
| Issue    | Serving content fails with error "Error while retrieving xsApp configuration". |
| Cause    | The HTML5 application repository is not available.                             |
| Solution | Wait until HTML5 application repository is restarted.                          |

## 500: Failed to use dynamic destination

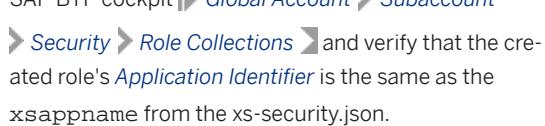
| Term     | Description   |
|----------|---|
| Issue    | Serving content from the application router failed with an internal server error. In the application router application log, an error: "Destination <destinationName> is not defined as a dynamic destination in destination service, configure additional property HTML5.DynamicDestination true" appears. |
| Cause    | The destination name provided on the host or path level is not defined as a dynamic destination in destination service.   |
| Solution | In the additional properties section of the destination section of the SAP BTP cockpit, add the <code>HTML5 . DynamicDestination</code> property and set the value to <code>true</code> .   |

## 500: Missing xs-app.json in HTML5 application repository when reading an HTML5 application file from the application router

|          |  |
|----------|--|
| Issue    | Serving content from the application router fails with error "Application does not have xs-app.json".          |
| Cause    | xs-app.json file is missing in HTML5 Application.  |
| Solution | Redeploy the HTML5 Application with xs-app.json file or ask the business service owner to add the xs-app.json. |

## Bearer token invalid

| Term  | Description   |
|-------|---|
| Issue | The <code>app-router</code> token exchange with a business service token fails. XSUAA returns the following error: "Bearer token invalid, requesting client does not have grant_type=user_token or no scopes were granted." |
| Cause | The <code>uaa.user</code> scope in XSUAA instance configuration is missing or the target business service is not subscribed.  |

| Term     | Description   |
|----------|---|
| Solution | <ol style="list-style-type: none"> <li>If missing, add the following to the <code>xs-security.json</code> file:<br/> <pre>"scopes": [ { "name": "uaa.user", "description": "UAA" } ], "role-templates": [ { "name": "Token_Exchange", "description": "UAA", "scope-references": [ "uaa.user" ] } ]</pre> </li> <li>Redeploy or update the XSUAA service instance.</li> <li>Make sure that the users have the new role_template, <code>Token_Exchange</code> added to their role_collections in the SAP BTP cockpit  and verify that the created role's <code>Application Identifier</code> is the same as the <code>xsappname</code> from the <code>xs-security.json</code>.</li> <li>If the roles of the target business service does not appear on the subscriber tenant UI, it means that the target business service subscription failed or it was bound to the application router after the subscription took place. In this case, try to unsubscribe and subscribe again. Make sure that the target business service roles appear on subscriber tenant UI.</li> </ol> |

## Caching issue in browser

| Term     | Description  |
|----------|--|
| Issue    | Your application does not work properly after logging out and when you try to log back in, for example, click anywhere on the application screen, nothing happens.   |
| Cause    | The main page of your application, that appears after you log in, is cached by the browser. Clicking links does not reach the backend (application router) and the log in process does not work.   |
| Solution | <p>Check that the main page is not configured to be cached by the browser in your <code>xs-app.json</code> file. The best practice is to model the <code>cacheControl</code> as follows:</p> <pre>{   "routes": [     {       "source": "^/ui/index.html",       "target": "index.html",       "service": "html5-apps-repo-rt",       "authenticationType": "xsuaa",       "cacheControl": "no-cache, no-store, must-revalidate"     }   ] }</pre> |

## Running application router after subscribing to it fails

| Term     | Description  |
|----------|--|
| Issue    | After subscribing to the application router using <subdomain>-<myapprouter>. <scp domain> format, it fails and returns the following error, "route not found."   |
| Cause    | <p>In order to support subscriptions out-of-the-box, create a route using * . &lt;custom-domain&gt; format and map it to the application router. The * host represents a wildcard that during runtime should be replaced by the actual subscriber sub-domain.</p> <p>Without a custom domain, this type of route (wildcard host) cannot be created, only a fixed host can be used. Without a custom domain, the route format will be &lt;subdomain&gt;-&lt;myapprouter&gt;. &lt;scp domain&gt;. It means that for each new subscriber you need to create a new route with the expected sub-domain.</p> |
| Solution | If you do not have a custom domain before subscribing to the application router, create a route with the expected subscriber sub-domain.   |

If the lists don't contain a solution for your issue, you can create an incident in the SAP Support Portal. For more information, see [Getting Support \[page 490\]](#).

## Related Information

[Getting Support \[page 490\]](#)

## 4.1.5 Consuming APIs

Discover APIs available on SAP BTP.

SAP BTP supports APIs that your applications can consume to interact, communicate, and exchange information with other applications, websites, and devices.

## SAP Business Accelerator Hub

The central repository for APIs from SAP and selected partners is the SAP Business Accelerator Hub. It contains reference documentation for a variety of REST and ODATA API packages that enable you to leverage the capabilities of SAP BTP.

For an overview of all API offerings for SAP BTP, see [SAP BTP on SAP Business Accelerator Hub](#).

If you're looking for information on APIs for a specific service, you can also use the [service catalog](#) in the SAP Discovery Center. Click on a service tile to display the available information. If the service has APIs, you can find links to their documentation under *Resources*. You can also check if a service is available in your region by opening the *Service Plan* tab.

## Related Information

[APIs \[page 128\]](#)

<https://answers.sap.com/tags/86cfb43d-204e-485e-a52d-41d9e73a5cd8>

<https://help.sap.com/docs/business-accelerator-hub/sap-business-accelerator-hub/what-is-sap-business-accelerator-hub?version=Cloud>

<https://blogs.sap.com/tags/86cfb43d-204e-485e-a52d-41d9e73a5cd8>

### 4.1.6 Adding Authentication and Authorization

Developers create authorization information for business users in their environment and deploy this information in an application. They make this available to administrators, who complete the authorization setup and assign the authorizations to business users.

Developers store authorization information as design-time role templates in the `xs-security.json` security descriptor file. Using the cockpit, administrators of the environment assign the authorizations to business users.

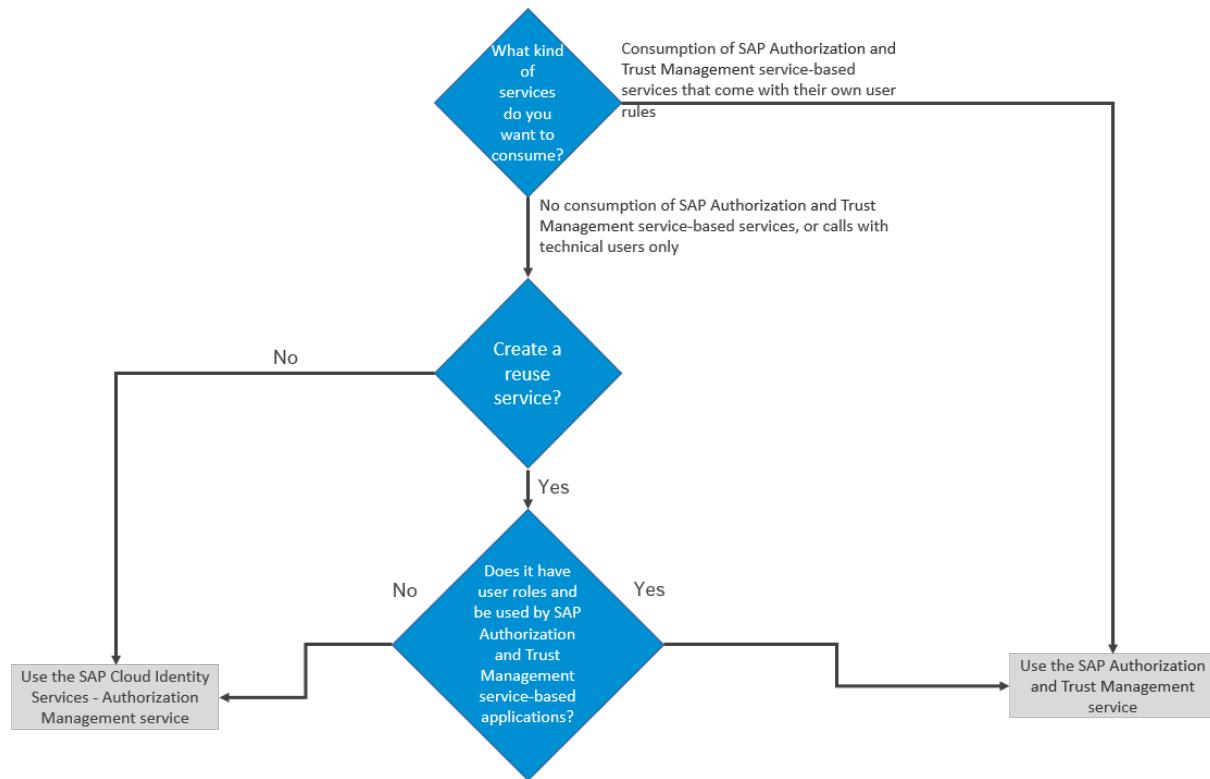
The following sections contain the process of adding authentication and authorization checks for protecting your applications, links to a number of associated tutorials, extended tasks for creating authorization artifacts, as well as reference information, including the syntax required to set the properties and values defined in the application security descriptor file.

### Using the SAP Authorization and Trust Management service or the SAP Cloud Identity Services

SAP BTP is currently replacing the authorization management done by the SAP Authorization and Trust Management service, with an integrated solution with the SAP Cloud Identity Services - Authorization Management service. This service is integrated into the SAP Cloud Identity Services, which will offer authentication, authorization, user provisioning and management in one place.

Currently, there are still use cases where the usage of the old model with the SAP Authorization and Trust Management service is simpler when it comes to the technical setup, especially in the administration for business users.

For decision considerations on when to use the SAP Authorization and Trust Management service and when to use the SAP Cloud Identity Services - Authorization Management service as the authorization management system on SAP BTP, you can follow this decision tree that will be regularly updated:



Developers use the Cloud Application Programming model for authorization policies. For more information, see [Authorization and Access Control](#).

## Protecting Your Application

Application security is maintained in the application security descriptor file (`xs-security.json`). You can use this file to define application-based security artifacts, which are the building blocks for authorizations. In this section, you can learn how to create this file and use it to create a service instance of the SAP Authorization and Trust Management service. Following this single tenant scenario, move on in increasing complexity to procedures that depict how to propagate user information between applications or services, add multitenancy to your application, and set up instance-based authorizations.

## Tutorials for Authentication and Authorizations

Use tutorials to get familiar with the SAP Authorization and Trust Management service in the Cloud Foundry environment of SAP BTP.

For more information, see [Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#).

## Tasks On Demand

Here you can find information regarding accessing the UAA admin APIs for creating XSUAA artifacts, as well as the process of configuring redirect URLs.

In addition, as an application developer, you may want to create role collections for immediate use. For example, to deliver role collections that administrators can use in the SAP BTP cockpit and easily assign to users in an onboarding process.

For more information, see [Create Role Collections with Predefined Roles \[page 537\]](#).

## Reference Information

For the syntax required to set the properties and values defined in the application security descriptor file (`xs-security.json`), see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

## Related Information

[SAP Authorization and Trust Management Service \[page 3023\]](#)

### 4.1.6.1 Protecting Your Application

Developers create authorization information for business users in their environment; this information is deployed in an application and made available to administrators who complete the authorization setup and assign the authorizations to business users.

## Lifecycle of Security Artifacts for Developers and Administrators

Developers store authorization information as design-time role templates in the application security descriptor file (`xs-security.json`). Using the `xsuaa` service broker, they deploy the security information to the `xsuaa` service. The administrators view the authorization information in role templates, which they use as part of the run-time configuration. They use the role templates to build roles, which are aggregated in role collections. The role collections are then assigned, in turn, to business users.

The tasks required to set up authorization artifacts are performed by two distinct user roles: the application developer and the administrator of the Cloud Foundry environment. After the deployment of the authorization artifacts as role templates, the administrator of the application uses the role templates provided by the developers for building role collections and assigning them to business users.

## ⓘ Note

To test authorization artifacts after deployment, developers can use the role templates to build role collections and assign authorization to business users in an authorization tool based on a REST API.

### Setting Up Authorization Artifacts (Developers)

| Step | Task   | User Role             | Tool   |
|------|--|-----------------------|--|
| 1    | Specify the application security descriptor file containing the functional authorization scopes for your application.  | Application developer | Text editor                                  |
|      | (If applicable) If you want to create an OAuth 2.0 client in an application-related subaccount, you must use a separate security descriptor file where you specify the subaccount. | Application developer | CF command line interface                    |
| 2    | Create role templates for the application using the application security descriptor file.  | Application developer | Text editor                                  |
| 3    | Create a service instance from the xsuaa service using the service broker  | Application developer | CF command line interface                    |
| 4    | Bind the service instance to the application by including it into the manifest file  | Application developer | Text editor                                  |
| 5    | Deploy the application   | Application developer | CF command line interface or SAP BTP cockpit |

## ⓘ Note

For more information about the different steps in setting up authorization artifacts as an application developer, refer to the associated procedures in this section.

[Add Authentication and Functional Authorization Checks to Your Application \[page 503\]](#)

[Propagate User Information Between Applications or Services \[page 508\]](#)

[Set Up Your Application for Multitenancy \[page 516\]](#)

[Setting Up Instance-Based Authorizations \[page 521\]](#)

## Setting Up Authorization Artifacts (Account Administrators)

| Task  | Links  |
|---|--|
| Assign the role collection to the users provided by an identity provider              | <a href="#">Working with Role Collections [page 2274]</a>                          |
| (If you do use a custom identity provider) Assign the role collections to user groups | <a href="#">Map Role Collections to User Groups [page 2281]</a>                    |
| Assign the role collections to users and user groups, manage attribute mappings       | <a href="#">Mapping Role Collections in the Subaccount [page 2281]</a>             |
| Create a role collection and assign roles to it                                       | <a href="#">Define a Role Collection [page 2275]</a>                               |
| Use an existing role or create a new one using role templates                         | <a href="#">Add Roles to Role Collections on the Application Level [page 2293]</a> |

## Related Information

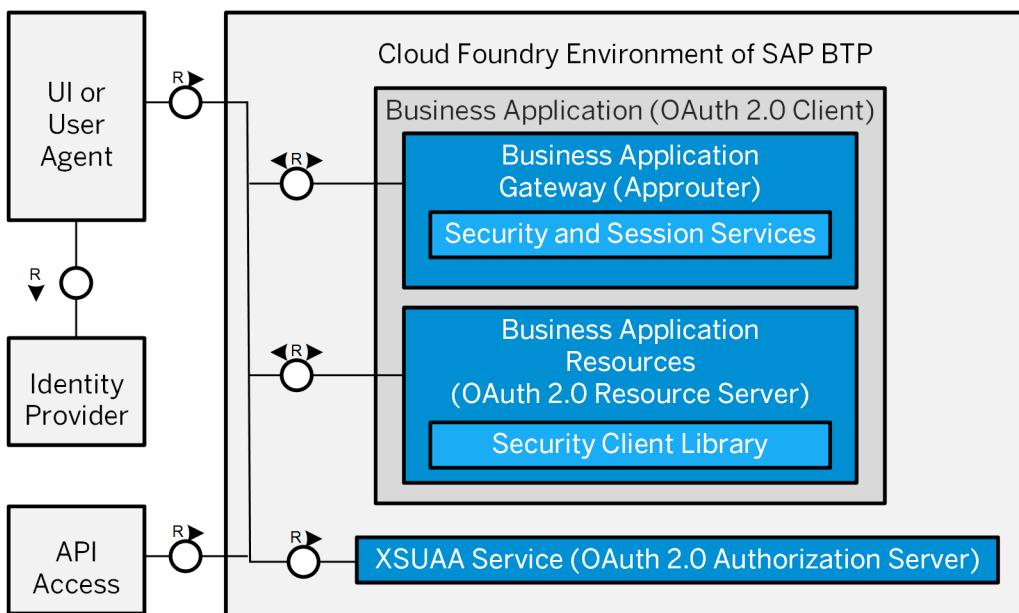
[Application Security Descriptor Configuration Syntax \[page 545\]](#)

### 4.1.6.1.1 Add Authentication and Functional Authorization Checks to Your Application

Learn how to create a security application descriptor file and use it to create a service instance of the Authorization and Trust Management service.

## Context

You want to protect your application resources (functionality and data) so that only authenticated and authorized business users are able to access these resources.



You declare your application security descriptor with JSON syntax and store it in a flat file on the filesystem. The standard name for this file is `xs-security.json`.

## Procedure

1. Create your application security descriptor (`xs-security.json`) and set the `xsappname` element.

The element `xsappname` defines a prefix for the runtime name of the application. For each tenant (subaccount), which has subscribed to the application, the XSUAA service supplies a unique tenant index for the application subscription and internally concatenates the tenant index with `xsappname` at runtime (see also next step).

### Sample Code

```
{
  "xsappname" : "hello-world",
  ...
}
```

2. Set the `scopes` element.

Scopes represent the API endpoints - or functions - for which access should be restricted. They're required if you want to define the functions a user is authorized to process. A scope has an arbitrary name and must be prefixed with the runtime application name to distinguish the user scopes between tenants (subaccounts) which have subscribed to the application and equally named scopes between different applications. The `$XSAPPNAME` dummy value is a wildcard for the application runtime name and is used to prefix the arbitrary scope names. The XSUAA service substitutes the `$XSAPPNAME` dummy value with the application runtime name for each of the subscribed tenants (see also step 1).

### Sample Code

```
{
```

```

...
"scopes" : [
  {
    "name" : "$XSAPPNAME.Display",
    "description" : "display" },
  {
    "name" : "$XSAPPNAME.Edit",
    "description" : "edit" },
  {
    "name" : "$XSAPPNAME.Delete",
    "description" : "delete" }
],
...
}

```

### 3. Set the `attributes` element.

Attributes represent the data entities for which access should be restricted. They're required if you want to define which data a user is authorized to process.

Attributes provide instance-based, fine-granular authorization checks. See [Setting Up Instance-Based Authorizations \[page 521\]](#).

#### ↔ Sample Code

```

{
...
"attributes" : [
  {
    "name" : "Country",
    "description" : "Country",
    "valueType" : "string" },
  {
    "name" : "CostCenter",
    "description" : "CostCenter",
    "valueType" : "int" }
],
...
}

```

### 4. Set the `role-templates` element.

Role templates combine scopes with attributes and serve as templates from which roles are created later by the administrator.

#### ↔ Sample Code

```

{
...
"role-templates" : [
  {
    "name" : "Viewer",
    "description" : "View all books",
    "scope-references" : [ "$XSAPPNAME.Display" ],
    "attribute-references" : [ "Country" ] },
  {
    "name" : "Editor",
    "description" : "Edit, delete books",
    "scope-references" : [ "$XSAPPNAME.Edit", "$XSAPPNAME.Delete" ],
    "attribute-references" : [ "Country", "CostCenter" ] }
]
}

```

5. Deploy the application security descriptor (`xs-security.json`).

Example:

#### ↔ Sample Code

```
cf create-service xsuaa application my_xsuaa_service_instance -c xs-security.json
```

6. Check the scopes in the application.

#### ↔ Sample Code

```
app.get('/books', checkReadScope, getBooks);
// Scope check
function checkReadScope(req, res, next) {
    if (req.authInfo.checkLocalScope('read')) {
        return next();
    } else {
        console.log('Missing the expected scope');
        res.status(403).end('Forbidden');
    }
}
```

The `checkReadScope` function ensures that only a user with the correct authorizations can look at the books.

See [Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#) for ways to check scopes in Java and other frameworks, like Spring.

7. In the manifest file, add the binding for the service instance to your application.

#### ↔ Sample Code

```
...
services:
- my_xsuaa_service_instance
```

8. Deploy the application.

#### ↔ Sample Code

```
cf push
```

## Results

You've declared and deployed your application security descriptor (`xs-security.json`) with scopes (functional authorizations), attributes (data authorizations), and role templates (to combine scopes with attributes). The `xsappname` element is mandatory. The `scopes`, `attributes`, and `role-templates` elements are optional. You've also bound your service instance to your application, and then deployed it.

## ↔ Sample Code

```
{  
  "xsappname" : "hello-world",  
  "scopes" : [  
    {  
      "name" : "$XSAPPNAME.Display",  
      "description" : "display" },  
    {  
      "name" : "$XSAPPNAME.Edit",  
      "description" : "edit" },  
    {  
      "name" : "$XSAPPNAME.Delete",  
      "description" : "delete" }  
  ],  
  "attributes" : [  
    {  
      "name" : "Country",  
      "description" : "Country",  
      "valueType" : "string" },  
    {  
      "name" : "CostCenter",  
      "description" : "CostCenter",  
      "valueType" : "int" }  
  ],  
  "role-templates" : [  
    {  
      "name" : "Viewer",  
      "description" : "View all books",  
      "scope-references" : [ "$XSAPPNAME.Display" ],  
      "attribute-references" : [ "Country" ] },  
    {  
      "name" : "Editor",  
      "description" : "Edit, delete books",  
      "scope-references" : [ "$XSAPPNAME.Edit", "$XSAPPNAME.Delete" ],  
      "attribute-references" : [ "Country", "CostCenter" ] }  
  ]  
}
```

## Next Steps

To implement web access using a browser or a browser-based user interface, you can use an application router to enable you to create a secure route to your application, or an open source UI framework, such as Spring (Boot). For more information, see the related links.

## Related Information

[Application Router \[page 409\]](#)

[Open Source UI Framework](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

## **4.1.6.1.2 Propagate User Information Between Applications or Services**

When a business application communicates with a service, you must decide whether you want to propagate the identity of the user that called the business application, or if a call from machine-to-machine is sufficient.

### **Principal Propagation Versus Technical Communication**

If the business application triggers an action of the service that should be auditable or requires that the identity of the user be known, use principal propagation. Principal propagation enables the identity of the user to be propagated from the business application to the service.

If the business application triggers an action of the service for which the identity of the user is unimportant, such as a regular clean-up task or the checking of a queue, then you can use technical communication. In technical communication, the service performs the action for the business application without knowing the identity of the user.

### **Cross-Application API Calls from an External System to Applications in the Cloud Foundry Environment**

If applications from an external system must make API calls to applications running in the Cloud Foundry environment, administrators must make sure that these applications can communicate with the relevant applications in the external system. In this case, the bearer assertion flow or client credentials identify the external application at the UAA, which can then issue a JSON web token. The external application can use this JSON web token when it makes the API calls to applications in the Cloud Foundry environment.

No browser is involved here. Users are propagated in the following ways:

- Using technical communication, for example, propagating scopes and authorities.
- Using a bearer assertion or client credentials (JSON web tokens) to propagate named users.

### **Related Information**

[Principal Propagation with Tightly Coupled Developments \[page 511\]](#)

[Technical Communication with Tightly Coupled Developments \[page 509\]](#)

[Principal Propagation \[page 3095\]](#)

## 4.1.6.1.2.1 Technical Communication with Tightly Coupled Developments

When a business application and a service are developed for the same subaccount, the two developments are tightly coupled together. The service is designed to be used with this particular application.

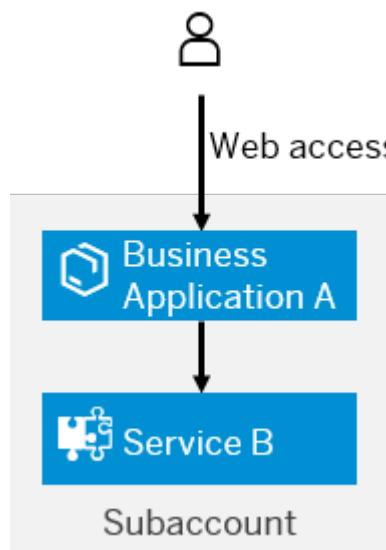
### Context

In this example, the call between the application and the service uses technical communication instead of the current user. The figure that follows illustrates this scenario.

#### ⓘ Note

With service, we do not mean a development that has been registered as a service with the Cloud Controller, but rather the role one application takes in relationship to another. Both developments can be applications, but one application takes on the role of a business application enabling business users to perform tasks. The other application plays the role of a service, enabling the development of the business application.

We also assume that both applications have their own application router. Having separate application routers means that the applications can define their own authorization models.



Two Developments Tightly Coupled Using Technical Communication

Like a trust arrangement, you must configure service B to grant the authority to business application A to use the `$XSAPPNAME.Create` scope. Likewise, you must configure business application A to accept such declarations of authority. This configuration is part of the `xs-security.json` of the UAA service instances of the two developments.

## Procedure

- As the service developer, declare the grant of authority to business application A in the `scopes` section of the `xs-security.json` for service B.

### Sample Code

```
{  
    "xsappname" : "servb",  
    "tenant-mode" : "shared",  
    "scopes" : [  
        {  
            "name" : "$XSAPPNAME.Create",  
            "description" : "create",  
            "grant-as-authority-to-apps" :  
                [ "$XSAPPNAME(application, appa)" ]  
        }  
    ]  
}
```

Under `scopes`, the scope `$XSAPPNAME.Create` grants authority to business application A to create data. Here the granted app is business application A and has been given the authorization to create data. When you create the UAA service instance for business application A, the system creates an OAuth client for business application A to call over to service B. When you bind the service, you make business application A aware of the client. In the security descriptor for the UAA service instance of business application A, accept this grant.

### Note

`$XSAPPNAME` is replaced at runtime with the application name. The parameters that follow `$XSAPPNAME` are the service plan for the XSUAA service and `xsappname` as defined in the `xs-security.json`. These parameters help uniquely identify other applications in the subaccount.

- Create the UAA service instance for service B.

For example:

```
cf create-service xsuaa application servb-uaa -c /servb/security/xs-security.json
```

- As the business application developer, accept the authority of other applications in the `authorities` section of the `xs-security.json` in business application A.

### Sample Code

```
{  
    "xsappname" : "appa",  
    "tenant-mode" : "shared",  
    "authorities" : [ "$ACCEPT_GRANTED_AUTHORITIES" ],  
}
```

In this example, business application A accepts all authorities granted by any other application. You can explicitly define which authorities you want business application to accept. For example:

```
"authorities" : [ "$XSAPPNAME(application, servb).Create" ].
```

### ⓘ Note

For the sake of this example, we've removed any scopes and role templates for application A. Naturally, we recommend that you protect application A with appropriate scopes and deliver role templates.

4. Create the UAA service instance for business application A.

For example:

```
cf create-service xsuaa application appa-uaa -c /appa/security/xs-security.json
```

5. Deploy the developments.
6. As an administrator, create a role from the role templates for business application A and assign the role to a user.

In the example, we created here, there are no role template to create roles from.

For more information, see [Administration: Managing Authentication and Authorization](#).

7. As an end user, access business application A.

When you access business application A, business application A accesses service B on your behalf.

Because service B granted the "\$XSAPPNAME.create scope to business application A, business application A is allowed to access the resource.

## 4.1.6.1.2.2 Principal Propagation with Tightly Coupled Developments

A scenario is tightly coupled when a business application calls a service within the same subaccount in the Cloud Foundry environment. The business application calls the service with principal propagation, meaning information about the current user is carried over with the service call.

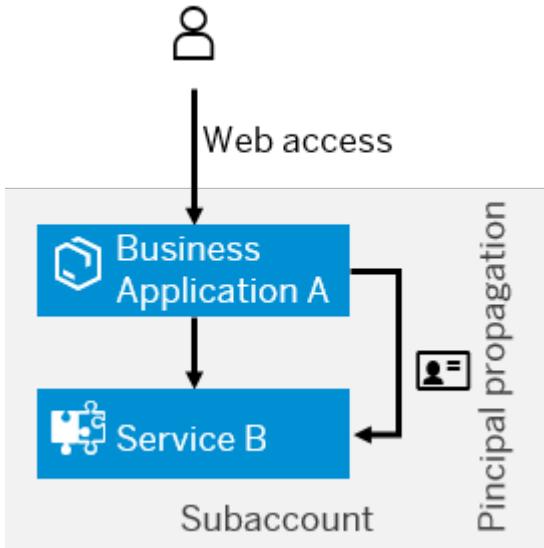
### Context

The figure that follows illustrates this scenario.

### ⓘ Note

With service, we do not mean a development that has been registered as a service with the Cloud Controller, but rather the role one application takes in relationship to another. Both developments can be applications, but one application takes on the role of a business application enabling business users to perform tasks. The other application plays the role of a service, enabling the development of the business application.

We also assume that both applications have their own application router. Having separate application routers means that the applications can define their own authorization models.



Two Applications, Tightly Coupled Using Principal Propagation

In this scenario, we enable the service to accept the JSON Web Token (JWT) of the business application. The JWT of the business application includes the scopes of both developments. When the service receives a JWT of the business application with the scopes of the service in it, the service implicitly accepts the JWT of the business application. This configuration is part of the `xs-security.json` of the UAA service instances of the two developments.

## Procedure

- As the service developer, add the attribute `granted-apps` to scopes you want to share with other business applications in the `scopes` section of the `xs-security.json` for this service.

In this attribute, list the names of the applications (`xsappname`) you want to share the scope with.

### Sample Code

```
"scopes": [
  {
    "name": "$XSAPPNAME.Viewer",
    "description": "View log",
    "granted-apps": ["$XSAPPNAME(application, appa)"]
  },
  {
    "name": "$XSAPPNAME.Editor",
    "description": "view and write logs",
  }
]
```

In this example, we have defined two scopes, `Viewer` and `Editor`. Under `granted-apps`, we list the application names with which we want to share the `Viewer` scope. All apps listed here can consume this scope in their role templates.

## ⓘ Note

`$XSAPPNAME` is replaced at runtime with the application name. The parameters that follow `$XSAPPNAME` are the service plan for the XSUAA service and `xsappname` as defined in the `xs-security.json`. These parameters help uniquely identify other applications in the subaccount.

You might reserve the `Editor` scope for a role template for the service. The editor role is only meant for administrators.

2. Create the UAA service instance for service B.

For example:

```
cf create-service xsuaa application servb-uaa -c /servb/security/xs-security.json
```

3. As the business application developer, add scope references to the scope of the service in the `role-templates` section of the `xs-security.json` in business application A.

## ⚡ Sample Code

```
"role-templates": [
  {
    "name": "Viewer",
    "description": "View all data of the service",
    "scope-references": [
      "$XSAPPNAME.Display",
      "$XSAPPNAME(application,servB).Viewer"
    ]
  }
]
```

Under `role-templates`, the `Viewer` template includes a reference to the `Viewer` scope of service B. This reference enables administrators to assign a single role to a user to grant authorizations for both applications.

4. Create the UAA service instance for application A.

For example:

```
cf create-service xsuaa application appa-uaa -c /appa/security/xs-security.json
```

5. Deploy the developments.
6. As an administrator, create a role from the role template from business application A and assign the role to a user.

Only assign the `Viewer` template from business application A to your test user and no templates from service B.

For more information, see [Administration: Managing Authentication and Authorization](#).

7. As an end user, test access business application A.

Your test user can use the scopes delivered from the `Viewer` template to access business application A or service B directly or indirectly.

## 4.1.6.1.2.2.1 Use Foreign Scope Option for Principal Propagation with Tightly Coupled Developments

With foreign scope reference, configure the business application to accept new scopes for the service without having to modify the business application.

### Procedure

- As the service developer, add the attribute `granted-apps` to scopes you want to share with other business applications in the `scopes` section of the `xs-security.json` for this service.

In this attribute, list the names of the applications (`xsappname`) you want to share the scope with.

#### ↔ Sample Code

```
"scopes": [
  {
    "name": "$XSAPPNAME.Read",
    "description": "read",
    "granted-apps": ["$XSAPPNAME(application, appa)"]
  },
  {
    "name": "$XSAPPNAME.Approve",
    "description": "approval"
  }
]
```

In this example, we've defined two scopes, `Read` and `Approve`. Under `granted-apps`, we list the application names with which we want to share the `Read` scope. All apps listed here can consume this scope in their role templates.

#### ⓘ Note

`$XSAPPNAME` is replaced at runtime with the application name. The parameters that follow `$XSAPPNAME` are the service plan for the XSUAA service and `xsappname` as defined in the `xs-security.json`. These parameters help uniquely identify other applications in the subaccount.

You might reserve the `Approve` scope for a role template for the service. The approver role is only meant for administrators.

- Create the UAA service instance for service B.

For example:

```
cf create-service xsuaa application servb-uaa -c /servb/security/xs-security.json
```

- In the business application, accept all foreign scope references.

#### ↔ Sample Code

```
{
```

```

"xsappname": "appa",
"tenant-mode": "shared",
"description": "Application ecurity descriptor for application A",
"foreign-scope-references": [ "$ACCEPT_GRANTED_SCOPES" ],
"scopes" : [
    {
        "name" : "$XSAPPNAME.View",
        "description" : "View data"
    }
],
"role-templates": [
    {
        "name" : "View",
        "description" : "View data",
        "scope-references" : [
            "$XSAPPNAME.View"
        ]
    }
]
}

```

Under `foreign-scope-references`, the application security descriptor accepts any and all foreign scopes that were granted to it. You can limit the references to just those applications you want to accept. For example, `"$XSAPPNAME(application,servb).Read`

4. Create the UAA service instance for application A.

For example:

```
cf create-service xsuaa application appa-uaa -c /appa/security/xs-security.json
```

5. Deploy the developments.
6. As an administrator, create a role from the role template from business application A and assign the role to a user.

Only assign the `Viewer` template from business application A to your test user and no templates from service B.

For more information, see [Administration: Managing Authentication and Authorization](#).

7. As a business user, test business application A.

Your test user can use the scopes delivered from the `Viewer` template to access business application A or service B directly or indirectly.

## Related Information

[Principal Propagation with Tightly Coupled Developments \[page 511\]](#)

### 4.1.6.1.3 Set Up Your Application for Multitenancy

Learn how to add multitenancy to your application and make it available for other subaccounts using the SaaS Provisioning service and the SAP Authorization and Trust Management service.

#### Context

You've created an application in your subaccount that is secured by the SAP Authorization and Trust Management service. You now want to make that application available to other subaccounts (tenants). You'll use the SaaS Provisioning service to make your application available to a consumer subaccount within your global trial account.

You declare your application security descriptor with JSON syntax and store it in a flat file on the filesystem. The standard name for this file is `xs-security.json`.

#### Procedure

1. Enable multitenancy in the application security descriptor file.
  - a. Go to the folder where the `xs-security.json` file is stored.
  - b. Change the value of the `tenant-mode` parameter to `shared`.
  - c. Under the `scopes` element, add access to the SaaS Provisioning service to call the product list callback API directly. You'll implement the callbacks in Step 3.

##### Sample Code

```
"scopes": [
    {
        "name": "$XSAPPNAME.read",
        "description": "With this scope, USER can read
products."
    },
    {
        "name": "$XSAPPNAME.Callback",
        "description": "With this scope set, the callbacks for
tenant onboarding, offboarding and getDependencies can be called.",
        "grant-as-authority-to-apps": [
            "$XSAPPNAME(application,sap-provisioning,tenant-
onboarding)"
        ]
    }
],
```

- d. Save the file.
2. Update the manifest.

In this step, you need to complete the following tasks:

- Add a new routing pattern.
- Add the service binding for the SaaS Provisioning service.

- a. Go to your application folder and open the `manifest.yml` file.
- b. For the application router, add the `TENANT_HOST_PATTERN` parameter under the `env` parameter.

This parameter specifies a generic route for all tenants to call the application over the approuter.

#### Sample Code

```
env:
  destinations: >
    [
      { "name": "hw-dest",
        "url": "https://product-list-ap25.cfapps.eu10.hana.ondemand.com",
        "forwardAuthToken": true}
    ]
  TENANT_HOST_PATTERN: "^(.*)-approuter-product-list-
ap25.cfapps.eu10.hana.ondemand.com"
```

#### Restriction

The value of the `TENANT_HOST_PATTERN` parameter must be in lowercase.

- c. Add the service binding for the SaaS Provisioning service to your application.

Adding the service binding of the SaaS Provisioning service in the `manifest.yml` file will automatically bind the service instance to your application, when deploying it.

#### Sample Code

```
services:
  - xsuaa-service-tutorial
  - saas-registry-tutorial
```

3. Implement the subscribe/unsubscribe endpoints.

To enable other subaccounts to subscribe to your application, you need to implement an endpoint for the SaaS registration manager to subscribe/unsubscribe.

- a. Go to the `myapp` folder and open the `index.js` file.
- b. Add the following lines of code after the `checkReadScope` function. (Replace the `ap25` string with the string that you used when deploying your application security model. Adapt the region code if your trial isn't in the `eu10` region.)

#### Sample Code

```
app.put('/callback/v1.0/tenants/*', function (req, res) {
  var consumerSubdomain = req.body.subscribedSubdomain;
  var tenantAppURL = "https:///" + consumerSubdomain + "-approuter-
product-list-ap25." + "cfapps.eu10.hana.ondemand.com/products";
  res.status(200).send(tenantAppURL);
});
app.delete('/callback/v1.0/tenants/*', function (req, res) {

  // Implement here any offboarding proces that's triggered when
  user deletes a subscription to the application
  res.sendStatus(200);
});
```

Return the HTTP 200 OK success status response if the offboarding process was successful.

If the offboarding failed, return the relevant error code and an accompanying message.

### ⓘ Note

It's highly recommended to use a response object that contains both the error code and a user-friendly, customer-oriented error message and technical details about the error in the following format:

```
{  
  "code":<The relevant response code, e.g., 500, 400>,  
  "messageForCustomer":<A customer-facing message. Please write in a  
  user-friendly format, as clearly as possible>,  
  "errorDetails":<Specify here additional technical details about the  
  error>  
}
```

- c. To be able to read the body of those calls, add the body parser module at line 9 of the `index.js` file.

### ↴ Sample Code

```
const bodyParser = require('body-parser')  
app.use(bodyParser.json())
```

- d. Add the body parser module as a dependency to the `product_list/myapp/package.json` file.

### ↴ Sample Code

```
"dependencies": {  
  "express": "^4.17.1",  
  "@sap/xsenv": "^2.2.0",  
  "@sap/xssec": "^3.0.0",  
  "passport": "^0.4.1",  
  "body-parser": "^1.19.0"  
}
```

4. Create a SaaS configuration file.

To make your multitenant application endpoints available for subscription to consumer subaccounts, you need to register the application in the Cloud Foundry environment by using the SaaS Provisioning service.

To register your application, you need a configuration file called `config.json`. In this file, you specify the subscription URL, the name, and description of your application. The `xsappname` must be the same as the `xsappname` in the `xs-security.json` file.

- a. Go to your application folder and create the `config.json` file.
- b. Insert the following lines.

### ↴ Sample Code

```
{  
  "xsappname": "product-list",  
  "appUrls": {  
    "onSubscription" : "https://product-list-  
ap25.cfapps.eu10.hana.ondemand.com/callback/v1.0/tenants/{tenantId}"  
  },  
  "displayName" : "Product List MTA",  
  "description" : "Product list MTA sample application",  
  "category" : "Custom SaaS Applications"
```

```
}
```

5. Delete the old service instance of the SAP Authorization and Trust Management service.

When you change the tenant mode from dedicated to shared like you did in step 1, it's not enough to update the service instance. You have to first unbind and delete the old service instance, to be able to later recreate it with the updated tenant mode settings.

- a. Unbind the existing service instance from your application.

```
cf unbind-service <APP_NAME> <SERVICE_INSTANCE>
```

#### ↔ Sample Code

```
cf unbind-service product-list xsuaa-service-tutorial
```

- b. Unbind the existing service instance from the application router.

```
cf unbind-service <APP_NAME> <SERVICE_INSTANCE>
```

#### ↔ Sample Code

```
cf unbind-service approuter xsuaa-service-tutorial
```

- c. Delete the existing service instance.

```
cf delete-service <SERVICE_INSTANCE>
```

#### ↔ Sample Code

```
cf delete-service xsuaa-service-tutorial
```

6. Create service instances and redeploy your application.

- a. Log in to your Cloud Foundry account with the Cloud Foundry CLI.
- b. Go to your application folder and create the service instance with the `xs-security.json` security descriptor file.

```
cf create-service <SERVICE> <PLAN> <SERVICE_INSTANCE> -c xs-security.json
```

#### ↔ Sample Code

```
cf create-service xsuaa application xsuaa-service-tutorial -c xs-security.json
```

- c. Create the SaaS Provisioning service instance with the `config.json` file.

### ↔ Sample Code

```
cf create-service <SERVICE> <PLAN> <SERVICE_INSTANCE> -c config.json
```

```
cf create-service saas-registry application saas-registry-tutorial -c config.json
```

- d. Redeploy the application with the updated `manifest.yml` file.

### ↔ Sample Code

```
cf push
```

7. Create a route for a consumer subaccount.

Make your application reachable for consumer subaccounts by adding a new route in the Cloud Foundry CLI. The route is composed of the subdomain of the subscribing subaccount and the `TENANT_HOST_PATTERN` of the application router that you defined in the `manifest.yml` file. You have to create a new route for every subaccount (tenant) that subscribes to the application.

- a. Log in to the Cloud Foundry account where the application is deployed with the Cloud Foundry CLI.
- b. Create a route for the consumer subaccount.

```
cf map-route <APP_NAME> <DOMAIN> --hostname <APPLICATION_HOSTNAME>
```

### ⓘ Note

The `APPLICATION_HOSTNAME` is the combined string of the subdomain ID of the consumer subaccount and the `TENANT_HOST_PATTERN` from the `manifest.yml` file.

### ↔ Sample Code

```
cf map-route approuter cfapps.eu10.hana.ondemand.com --hostname consumer-tenant-ap25-approuter-product-list-ap25
```

8. Access the application with the consumer subaccount.

To access the application you need to subscribe to it. Follow these steps to subscribe to the SaaS application with the consumer subaccount and call the application URL.

- a. Open the SAP BTP trial.
- b. Navigate to your consumer subaccount.
- c. Choose *Subscriptions*.
- d. Choose *Product List MTA*.
- e. Choose *Subscribe*.
- f. Choose *Go to Application*.

You'll now see the application with the message no data because you have to assign the role collection to your user in the consumer subaccount.

9. Assign the role collection.

Assign the `ProductListViewer` role collection to your user. This role collection contains the necessary role to view the products in your application.

- a. Open the SAP BTP cockpit.
- b. Navigate to your consumer subaccount.
- c. Choose **Role Collections**.
- d. Choose the `ProductListViewer` role collection.
- e. Use the chevron at the right side to expand the role collection.
- f. Go to the `Users` section and choose `Edit`.
- g. Enter the e-mail address of the user that you want to assign to the role collection. Take care that your user's identity provider is SAP ID service.
- h. Save your changes.
- i. Clear your cache and reload the application URL.

#### Sample Code

```
https://consumer-tenant-ap25-approuter-product-list-
ap25.cfapps.eu10.hana.ondemand.com/products
```

The application will now show you the products.

## Related Information

[Add Multitenancy to a Node.js Application Secured by the SAP Authorization and Trust Management service](#)

### 4.1.6.1.4 Setting Up Instance-Based Authorizations

Instance-based authorizations are authorization checks based on the value of some authorization-relevant attribute defined by the application. Functional authorizations define whether you've permission to edit or view some function in the application, while instance-based authorizations define what data you can act upon.

As a developer, you develop your application as you did in our previous examples for functional authorization checks. In that example, we defined attributes in the application security descriptor (`xs-security.json`) even though we didn't use them. In that example, we defined `CostCenter` as an attribute. Let's look at an example.

Your application is used by a controller to view the expenses of their organization. So, your application shows the controller all invoices, which belong to the cost center of the controller. So, your application takes the dataset of invoices and shows the instance of that dataset where the cost center equals the cost center of the controller. Control access to this dataset by creating a role collection that includes the `CostCenter` attribute and the value required by the controller. Then assign that role collection to the user of the controller.

For more information about defining attributes in the application security descriptor, see [Add Authentication and Functional Authorization Checks to Your Application \[page 503\]](#).

For instance-based authorizations, you build checks against these attributes into your coding. To help you understand authorization checks against attributes, check out our sample bulletin board application on Github.

For more information, see <https://github.com/SAP-samples/cloud-application-security-sample/tree/master/spring-security-basis>.

Once you've built in your authorization checks, you've a few ways to populate the attributes with values, for your code to check.

## How Attribute Values Are Set

Once you've defined attributes in your role templates, how are attribute values filled? There are a couple different ways.

- Predefine the attribute values in role collections you deliver with your app.  
As a developer, you can define preconfigured role collections in your application security descriptor. When the role administrator assigns these role collections to users, the attribute values are hard-coded with the default values according to the role templates you define. These role collections enable a role administrator to get their users started quickly.
- The administrator builds static roles from your role templates.  
The administrator builds the roles they need, based on the role templates you deliver. The administrator builds each role, setting the values for the attributes you defined in the role templates. The administrator can't add or subtract any attributes. The administrator can only set the values. The administrator then bundles these roles into role collections and assigns them to users.
- The administrator builds dynamic roles from your role templates.  
The administrator builds the roles they need, based on the role templates you deliver. The administrator can decide to populate role values dynamically from the identity provider. When users log on, they receive a token with attributes defined by the identity provider. The manager of the identity provider controls the values a user has for given attributes, not the role administrator of the subaccount. This setup means that you as an application developer need to understand what kinds of attributes are available to the role administrator from the identity provider.

## Related Information

[Attributes \[page 2288\]](#)

### 4.1.6.1.5 Retrieving Access Tokens with Mutual Transport Layer Security (mTLS)

Mutual Transport Layer Security (mTLS) is considered more secure than the combination of client ID and client secret. Unlike retrieving the access token with client ID and client secret, no secret is shared between calling application and the service instance of SAP Authorization and Trust Management service (XSUAA).

This configuration enables your application to retrieve or exchange access tokens from an instance of the SAP Authorization and Trust Management service with mTLS. When using TLS, the client verifies the identity of the OAuth server during their handshake. By using mTLS, the OAuth server also verifies the identity of the client. Calls to other services and applications with the access token use the standard OAuth protocols.

You, as a developer, have the option to have the service provide the X.509 certificate for the binding or service key or if you already have your own public key infrastructure (PKI), you can provide your own.

### **Restriction**

If you use your own PKI, your certificates must be issued by certificate authorities trusted by the service.

For more information, see [Binding Parameters of SAP Authorization and Trust Management Service \[page 2301\]](#).

## **Related Information**

[Enable mTLS Authentication to SAP Authorization and Trust Management Service for Your Application \[page 523\]](#)

[Implementing Custom Token Retrieval from SAP Authorization and Trust Management Service with mTLS \[page 525\]](#)

### **4.1.6.1.5.1 Enable mTLS Authentication to SAP Authorization and Trust Management Service for Your Application**

Enabling your application to retrieve access tokens from SAP Authorization and Trust Management service(XSUAA) over mutual transport layer security (mTLS) requires you to enable mTLS token retrieval for the service instance. Then ensure the binding includes the X.509 credentials.

## **Prerequisites**

To handle tokens, use:

- Application router version 10.7.1 or later  
For more information, see <https://www.npmjs.com/package/@sap/approuter>.
- Our security libraries
  - For Java or Spring, see [XSUAA Token Client and Token Flow API](#) on *Github*.
  - For Python, see [SAP Python Security Library](#) on *Python Package Index*.
  - For Node.js, see [@sap/xssec](#) on *Node Package Manager (NPM)*.
- Your own integration  
For more information, see [Implementing Custom Token Retrieval from SAP Authorization and Trust Management Service with mTLS \[page 525\]](#).

### **Recommendation**

Build automation into your deployment or CI/CD pipeline. Client certificates have a relatively short lifetime. The default lifetime for certificates managed by the service is 7 days. Only by automating the process, can you ensure that credentials are rotated and distributed in a timely manner. Otherwise, you risk authentication failures from expired certificates.

## Context

Use the same procedure as outlined in [Add Authentication and Functional Authorization Checks to Your Application \[page 503\]](#). Modify this procedure with the following steps.

Be mindful of whether you're updating an existing application or starting with a brand new development.

- If you're adding mTLS to an existing application, add mTLS support to the credential types of the service instance. Then update all your bindings and service keys before removing the support for other credential types.

### → Tip

For an example of how to migrate from one secret type to another, see [Migrate from Instance Secrets to Binding Secrets \[page 2299\]](#).

- If you're starting from scratch, you can build in mTLS support from the beginning, without worrying about disrupting existing bindings or service keys.

## Procedure

1. Enable mTLS for the service instance.

Add the credential type x509 to the OAuth configuration of the service instance.

- The following is an example of the OAuth configuration from an application security descriptor (`xs-security.json`).

```
"oauth2-configuration": {  
    "credential-types": [ "binding-secret", "x509" ]  
}
```

In this example, binding secrets are the default credential type, but X.509 secrets are supported.

- The following is an example of the service as a resource in a multitarget application (MTA) descriptor (`mta.yaml`).

```
resources:  
- name: myXSUAA  
  type: com.sap.xs.uaa  
parameters:  
  service-plan: application  
  config:  
    xsappname: mAapp  
    oauth2-configuration:  
      credential-types:  
        - x509  
        - binding-secret
```

In this example, X.509 secrets are the default credential type. When deployed, any preconfigured bindings get service-managed client certificates. After deployment, you can create additional bindings with binding secrets.

2. Deploy the application or update the service instance.
3. Create bindings or service keys with the X.509 credentials.

You create the binding from the Cloud Foundry command-line interface (cf CLI),

- From the command-line interface, provide a `parameters.json` to configure the certificates managed by the service or to provide your own X.509 certificate.

The following is an example of a `parameters.json` with the certificates managed by the service.

```
{  
  "credential-type": "x509",  
  "x509": {  
    "key-length": 2048,  
    "validity": 8,  
    "validity-type": "DAYS"  
  }  
}
```

The following is an example of a `parameters.json` with certificates you provide.

```
{  
  "credential-type": "x509",  
  "x509": {  
    "certificate": "-----BEGIN CERTIFICATE-----...-----END  
CERTIFICATE-----",  
    "certificate-pinning": true  
  }  
}
```

Reference the `parameters.json` when you bind the service. For example:

```
cf bind-service myApp myXSUAA -c parameters.json
```

## Related Information

[Parameters for X.509 Certificates Managed by SAP Authorization and Trust Management Service \[page 2303\]](#)

[Parameters for Self-Managed X.509 Certificates \[page 2304\]](#)

[Rotate X.509 Secrets \[page 2298\]](#)

## 4.1.6.1.5.2 Implementing Custom Token Retrieval from SAP Authorization and Trust Management Service with mTLS

The application router and other token client libraries take care of getting tokens from the SAP Authorization and Trust Management service for you. If you build your own integration, you must handle token retrieval yourself.

### ⓘ Note

The service doesn't check for certificate revocation. To stop the service from accepting a certificate that is still valid, delete the relevant bindings or service keys. As soon as the binding is deleted, the service stops accepting the certificate.

For bindings with self-managed certificates and the `certificate-pinning` parameter set to `false`, you can rotate the secrets without deleting bindings. Just use a new certificate with the same subject and issuer distinguished name (DN). The service saves the new validity date of the new certificate.

For more information, see [Parameters for Self-Managed X.509 Certificates \[page 2304\]](#).

## → Recommendation

Build automation into your deployment or CI/CD pipeline. Client certificates have a relatively short lifetime. The default lifetime for certificates managed by the service is 7 days. Only by automating the process, can you ensure that credentials are rotated and distributed in a timely manner. Otherwise, you risk authentication failures from expired certificates.

When called with mTLS, the token endpoint is a little different from the standard endpoint. The path includes `authentication.cert` instead of just `authentication`.

`https://<subdomain>.authentication.cert.<landscape>/oauth/token`

For example:

`https://test-me.authentication.cert.eu20.hana.ondemand.com/oauth/token`

Use the `/well-known/openid-configuration` endpoint to find the endpoints for your subaccount. The endpoints appear under the `mtls_endpoint_aliases` parameter.

For example: `https://test-me.authentication.eu20.hana.ondemand.com/.well-known/openid-configuration`

The following is an example of part of the JSON returned by the endpoint.

```
{  
  ...  
  "mtls_endpoint_aliases": {  
    "token_endpoint": "https://test-  
me.authentication.cert.eu20.hana.ondemand.com/oauth/token",  
    "authorization_endpoint": "https://test-  
me.authentication.cert.eu20.hana.ondemand.com/oauth/authorize"  
  },  
  ...  
}
```

For X.509 authentication, the `grant_type` are as with client secret authentication:

- For UI user flows: Authorization code
- For API user flows: JWT bearer, refresh token, and password grant
- For technical flows: Client credentials

## ⓘ Note

Leave out the `client_secret` parameter. The `client_id` is still required.

Finally, when building the call, include the client certificate (public key) and private key for signing in your request. The following syntax is for CURL:

```
curl --cert <Path_Client_Cert> --key <Path_Private_Key>  
-XPOST https://<subdomain>.authentication.cert.<landscape>/oauth/token -d  
'grant_type=<Grant_Type>&client_id=<Client_ID>'
```

The following is an example in CURL:

```
curl --cert x509certificate.pem --key x509privatekey.pem  
-XPOST https://test-me.authentication.cert.eu20.hana.ondemand.com/oauth/  
token -d 'grant_type=client_credentials&client_id=sb-na-  
edb111d1-2c22-3eca-444-869fde307ac7!a5017'
```

## Related Information

[Enable mTLS Authentication to SAP Authorization and Trust Management Service for Your Application \[page 523\]](#)

### 4.1.6.2 Tutorials for the SAP Authorization and Trust Management Service

Follow the tutorials below to get familiar with the SAP Authorization and Trust Management service in the Cloud Foundry environment of SAP BTP.

| Tutorials for the SAP Authorization and Trust Management service in the Cloud Foundry environment  | Language / Framework | Link   |
|--|----------------------|--|
| Learn how to secure a basic single-tenant Node.js application. Start with a Node.js application that uses the express framework and SAPUI5 to display a list of products and add the security components step by step. | Node.js              | <a href="#">SAP Developers</a>  |
| Learn how to secure a basic java application. This tutorial starts with a Hello World Java application built with SAP Cloud SDK.   | Java, SAP Cloud SDK  | <a href="#">SAP Developers</a>  |
| Learn how to secure microservices in SAP BTP using spring-xsuaa and Spring security. Furthermore, learn how to test the secured application using the java-security-test utilities.                                    | Spring (Boot)        | <a href="#">GitHub</a>          |
| Learn how to add multitenancy to a node.js application and make it available for other subaccounts using the SaaS Provisioning service and the XSUAA.  | Node.js              | <a href="#">SAP Developers</a>  |

**Tutorials for the SAP Authorization  
and Trust Management service in the  
Cloud Foundry environment**

| Language / Framework  | Link  |
|---|---|
| Learn how to secure microservices in SAP BTP. This sample provides J2EE Configuration using web.xml and uses the SAP Java Buildpack.  | J2EE, SAP Java Buildpack<br><a href="#">GitHub (SAP Java Buildpack version &lt;= 1.26.0)</a> ↗<br><a href="#">GitHub (SAP Java Buildpack version &gt;=1.26.1)</a> ↗ |
| Learn how to build a cloud-native Node.js application that features secured service-to-service communication. The application shows you two different ways of securing service-to-service-communication (by propagating a business user or using a technical user). | Node.js<br><a href="#">GitHub</a> ↗   |
| Learn how to use the java-security library to perform JWT Validation as part of your Java application. Furthermore, learn how to test the secured application using the java-security-test utilities.   | Java<br><a href="#">GitHub</a> ↗  |
| Learn in this reference application how the service fits into a complete architecture of microservices that interact with each other propagating user information.  | Java<br><a href="#">GitHub</a> ↗  |
| Learn how to validate OAuth tokens using a Python library. Use this library to add authentication in your Python application.   | Python<br><a href="#">GitHub</a> ↗  |

**ⓘ Note**

This library isn't part of an SAP BTP license. However, it belongs to a related open source project.

## 4.1.6.3 Tasks On Demand

Here you can find information regarding the available extended tasks for creating authorization artifacts, as well as the process of configuring redirect URLs.

In addition, as an application developer, you may want to create role collections for immediate use. For example, to deliver role collections that administrators can use in the SAP BTP cockpit and easily assign to users in an onboarding process.

### [Update a Service Instance \[page 530\]](#)

You can update a service instance from the `xsuaa` service using the service broker.

### [Sharing and Unsharing Service Instances \[page 530\]](#)

You can share service instances of the SAP Authorization and Trust Management service across multiple spaces or environments. This has the advantage that you don't need to create separate service instances in multiple spaces or environments.

### [Compatible Changes in the Security Descriptor File \[page 531\]](#)

If you update a service instance, you can add, change, and/or delete scopes, attributes, and role templates. Whenever you change role templates, you adapt roles that are derived from the changed role template.

### [Retrieve Credentials for Remote Applications \[page 532\]](#)

You need to get the credentials of a service instance for an application that runs outside SAP BTP.

### [Configure Redirect URLs for Browser Logout \[page 535\]](#)

To avoid open redirect attacks, direct users to a safe and valid URL when they log out.

### [Create Role Collections with Predefined Roles \[page 537\]](#)

As an application developer, you want to create role collections for immediate use. You want to deliver role collections that administrators can use in the cockpit, and easily assign to users, for example in an onboarding process.

### [Validation and Revocation of Access Tokens \[page 539\]](#)

To confirm that access tokens were truly issued by the SAP Authorization and Trust Management service and are still valid, applications check the tokens for validity.

### [Include Tokens from Corporate Identity Providers or SAP Cloud Identity Services in Tokens of the SAP Authorization and Trust Management Service \[page 540\]](#)

In scenarios where your SAP BTP application interacts with applications that don't support tokens issued by the SAP Authorization and Trust Management service, but require a token from a corporate identity provider or SAP Cloud Identity Services, you can configure SAP Authorization and Trust Management service to include this token.

### [Best Practices When Handling Access Tokens \[page 542\]](#)

Calls to SAP Authorization and Trust Management service can fail for many reasons, including network latency, connection errors, service downtime, idle connections closed by routing components and so on. The following sections provide recommendations on when to retry connections and the caching of tokens.

### 4.1.6.3.1 Update a Service Instance

You can update a service instance from the xsuaa service using the service broker.

#### Context

You are running a service instance that grants user access to an application. It uses the security descriptor file `xs-security.json`. If you change properties, for example, you want to reflect the compatible changes you made in the `xs-security.json` file in an existing service instance.

##### ⚠ Restriction

If you define or update role templates and attributes in the `xs-security.json` file, observe the following configuration restrictions in [Relationship Between default-values of attribute-references and valueRequired](#).

#### Procedure

1. Edit the `xs-security.json` file and make your changes in the security descriptors.
2. Update the service instance. During the update, you use the security descriptors you changed in the `xs-security.json` file.

```
cf update-service <service_instance_name> -c xs-security.json
```

##### ❖ Example

```
cf update-service authorizationtest-uaa -c xs-security.json
```

### 4.1.6.3.2 Sharing and Unsharing Service Instances

You can share service instances of the SAP Authorization and Trust Management service across multiple spaces or environments. This has the advantage that you don't need to create separate service instances in multiple spaces or environments.

- You must have the [Subaccount Service Administrator](#) role collection in the specified subaccount. Use the SAP BTP cockpit to assign this role collection (see [Assigning Role Collections to Users or User Groups \[page 2277\]](#).)

Using the SAP BTP command line interface (btp CLI), you can share service instances across spaces or environments (for example, Cloud Foundry and Kubernetes). Shared service instances don't have the same credential. The client ID can be the same or different. Sharing service instances is possible with service instances of the application and broker plan.

#### Note

It depends on the specific service whether sharing and unsharing service instances is available.

#### Restriction

You can only share service instances in the same subaccount.

For more information, see [btp share services/instance](#) and [btp unshare services/instance](#).

### 4.1.6.3.3 Compatible Changes in the Security Descriptor File

If you update a service instance, you can add, change, and/or delete scopes, attributes, and role templates. Whenever you change role templates, you adapt roles that are derived from the changed role template.

Compatible Changes in xs-security.json

| Security Artifacts in the xs-security.json File | Action  |
|---|---|
| Scope   | <ul style="list-style-type: none"><li>• Add</li><li>• Delete</li><li>• Change<ul style="list-style-type: none"><li>• description</li><li>• granted-apps</li></ul></li></ul> |
| Attribute                                       | <ul style="list-style-type: none"><li>• Add</li><li>• Delete</li><li>• Change<ul style="list-style-type: none"><li>• description</li></ul></li></ul>                        |

- Add
- Delete
- Change
  - description
  - granted-apps

| Attribute | Action  |
|-----------|---|
| Scope     | <ul style="list-style-type: none"><li>• Add</li><li>• Delete</li><li>• Change<ul style="list-style-type: none"><li>• description</li><li>• granted-apps</li></ul></li></ul> |
| Attribute | <ul style="list-style-type: none"><li>• Add</li><li>• Delete</li><li>• Change<ul style="list-style-type: none"><li>• description</li></ul></li></ul>                        |

- Add
- Delete
- Change
  - description

#### Note

Do not change the `valueType` of the attribute.

| Security Artifacts in the xs-security.json File | Action   |
|---|--|
| Role template                                   | <ul style="list-style-type: none"> <li>• Add</li> <li>• Delete</li> <li>• Change <ul style="list-style-type: none"> <li>• description</li> <li>• scope-references</li> <li>• attribute-references (can be deleted only)</li> </ul> </li> </ul> |

**⚠ Restriction**

When you define or update role templates and attributes, observe the following configuration restrictions in [Relationship Between default-values of attribute-references and valueRequired](#).

#### 4.1.6.3.4 Retrieve Credentials for Remote Applications

You need to get the credentials of a service instance for an application that runs outside SAP BTP.

#### Prerequisites

- A service instance for your application is available.

**ⓘ Note**

For mutual transport layer security (mTLS), the application security descriptor supports the `binding-secret` and `x509` credential types by default. For the creation of a service key that enables the `x509` credential type, you need to pass a parameter file. See [Binding Parameters of SAP Authorization and Trust Management Service \[page 2301\]](#).

#### Context

**ⓘ Note**

Applications that run inside the service manager instance at SAP BTP get their credentials after the respective service has been bound to the application. However, you can't use binding for an application that runs outside of the service manager instance at SAP BTP.

Instead, you use a service key that is created in the service instance of the remote application. You need to get the credentials of the service instance for the remote application. The UAA service broker manages all

credentials, including those of the remote applications. The credentials you need are the OAuth 2.0 client ID and the client secret, or X.509 certificates.

First you generate a service key for the service instance of the remote application to enable access to the UAA service broker. Then you retrieve the generated service key with the credentials, including the OAuth 2.0 client ID and the client secret or X.509 certificates, from the UAA service broker. The remote application stores the service key. The application can now use this service key with the credentials (OAuth 2.0 client ID and the client secret of the remote application).

#### → Remember

Rotate credentials before they expire or, if they don't expire, rotate them regularly (see [Security Recommendations for SAP Authorization and Trust Management Service \[page 3081\]](#)).

## Procedure

1. Create a service key for the service instance of the remote application. If you want to use mutual transport layer security (mTLS), you may need to request it in a separate binding configuration file (see [Enable mTLS Authentication to SAP Authorization and Trust Management Service for Your Application \[page 523\]](#) and [Binding Parameters of SAP Authorization and Trust Management Service \[page 2301\]](#)).

```
cf create-service-key <service_instance_name> <service-key-name>
```

#### ❖ Example

```
cf create-service-key rem-app-service rem-app-sk
```

(For mTLS) `cf create-service-key <service_instance_name> <service-key-name> -c <parameter_file>`

#### ❖ Example

```
cf create-service-key rem-app-service rem-app-sk -c parameters.json
```

The `parameters.json` file can have the following content for the usage of XSUAA-managed certificates:

#### ↔ Sample Code

```
{  
  "credential-type": "x509"  
}
```

2. You want to retrieve the credentials including the OAuth 2.0 client ID and the client secret for the service instance of your remote application. Use the following command:

```
cf service-key <service_instance_name> <service_key_name>
```

#### ❖ Example

```
cf service-key rem-app-service rem-app-sk
```

Output with default binding-secret credential type:

#### ↔ Output Code

```
{  
    "apiurl": "https://api.authentication.corpdcen.acme.ondemand.com",  
    "clientid" : "sb-sample-app!t1",  
    "clientsecret" : "<client_secret>",  
    "credential-type": "binding-secret",  
    "identityzone" : "uaa",  
    "serviceInstanceId": "cbb63529-d725-4326-880c-92cbcd92",  
    "subaccountid": "194e6a6d-c590-5030-98b3-1baa6d8fcda4",  
    "tenantid": "095e5a7f-c480-5030-98b3-1cbb6e8fdfb4",  
    "uaadomain": "authentication.corpdcen.acme.ondemand.com",  
    "url" : "https://host.acme.com:40654/uaa-security",  
    "xsappname" : "sample-app!t1",  
    "..."  
}
```

(For mTLS) Output with x.509 credential type:

#### ↔ Sample Code

```
{  
    "credentials": {  
        "apiurl": "https://api.authentication.corpdcen.acme.ondemand.com",  
        "certificate": "-----BEGIN CERTIFICATE-----  
nMIIFvDCCA6...KJu+8fcIaUp7MVBIVZ\n-----END CERTIFICATE-----\n",  
        "certificate-pinning": true,  
        "certurl": "https://mysubaccount-  
hmq7fre.a.authentication.cert.corpdcen.acme.ondemand.com",  
        "clientid": "sb-x5app!t75135",  
        "credential-type": "x509",  
        "key": "-----BEGIN RSA PRIVATE KEY-----<private_key>\n-----END RSA  
PRIVATE KEY-----\n",  
        "serviceInstanceId": "cbb52529-d614-4326-880c-92ba937bcc92",  
        "subaccountid": "<subaccount_ID>",  
        "tenantid": "194e6b7e-c370-4941-98b3-1cbb6e8fdfb4",  
        "uaadomain": "authentication.corpdcen.acme.ondemand.com",  
        "url": "https://<subaccount>-  
hmq7fre.a.authentication.corpdcen.acme.ondemand.com",  
        "xsappname": "x5app!t75135",  
        "..."  
    }  
}
```

If your service needs an X.509 certificate and private key for mutual transport layer security (mTLS) in a single privacy enhanced mail (PEM) file, see [Extract Certificates for Mutual Transport Layer Security](#).

## 4.1.6.3.5 Configure Redirect URLs for Browser Logout

To avoid open redirect attacks, direct users to a safe and valid URL when they log out.

### Prerequisites

- You've an application that is using the application router.
- Use application router 5.7.0 or higher. For more information, see the related link.

### Context

You've deployed an application in the Cloud Foundry environment of SAP BTP. Business users log in and out of the application. As an application developer, you want to set a redirect URI, which directs users to a specific page once they've logged out. This page can be a logout page or your organization's employee portal. For security reasons, there's a check of this redirect URI.

- In the application descriptor (`xs-app.json`), you define a public logout page.
- To avoid redirects, for example to malicious web sites, `xsuaa` compares this public URL with a list of allowed redirect URLs. You define them in the OAuth 2.0 configuration property of the application security descriptor (`xs-security.json`).

### Procedure

1. Define `xs-app.json` in the application router folder of your application. Include a logout endpoint and define a logout page (here `logout.html`) which can be accessed without authentication.

#### ↔ Sample Code

```
{  
    "welcomeFile": "index.html",  
    "authenticationMethod": "route",  
    "logout": {  
        "logoutEndpoint": "/my/logout",  
        "logoutPage": "logout.html"  
    },  
    "routes": [  
        {  
            "source": "^/hw/",  
            "target": "/",  
            "destination": "hw-dest",  
            "scope": {  
                "GET": [  
                    "$XSAPPNAME.Display",  
                    "$XSAPPNAME.Update"  
                ],  
                "default": "$XSAPPNAME.Update"  
            }  
        }  
    ]  
}
```

```

        "source": "/index.html",
        "localDir": "resources",
        "cacheControl": "no-cache, no-store, must-revalidate"
    },
    {
        "source": "/logout.html",
        "localDir": "resources",
        "authenticationType": "none"
    },
    {
        "source": "^/(.*)",
        "localDir": "resources"
    }
]
}

```

2. Open a command prompt.
3. Log in to your Cloud Foundry environment using the Cloud Foundry command-line interface (cf CLI).
4. Choose your subaccount.

#### ⓘ Note

The cf CLI prompts you to choose an org. To find the org of your subaccount, use the cockpit to go to your subaccount. You can find the org in the Cloud Foundry tile under *Organization* (see [Navigate to Orgs and Spaces \[page 2433\]](#)).

5. Choose the space where your application is located.
6. Deploy the application to your space.
7. To define the redirect URIs, go to the folder where xs-security.json is stored.
8. Define the redirect URIs in the oauth2-configuration property.
9. Add a URL that contains the hostname of your application.

#### ⚡ Sample Code

```
{
  "xsappname": "<application_name>",
  "tenant-mode": "dedicated",
  "description": "My Sample Application with Application Router",
  "oauth2-configuration": {
    "token-validity": 900,
    "redirect-uris": ["https://<application_hostname>.<landscape_domain>/**"],
    "autoapprove": "true"
  }
}
```

#### ⓘ Note

The application domain is the first part of the application URL: For more information, see the related link.

#### ⚡ Example

```
myapplication.cfapps.hana.ondemand.com/**
```

### ❖ Example

For China (Shanghai) region:

```
myapplication.mycustomdomain.cn/**
```

10. Update the xsuaa service instance that is bound to your application.

```
cf update-service <service_instance> -c xs-security.json
```

### ⓘ Note

If the service instance isn't available, create it, bind your application to the service instance (see the related links), and restage the application.

You've implemented a valid redirect after logout, and xsuaa checks the redirect URL against the list of allowed URIs in the OAuth 2.0 configuration.

## Related Information

[Application Router \[page 409\]](#)

[Configuring Application URLs](#)

[Update a Service Instance \[page 530\]](#)

[Cloud Foundry Command Line Interface ↗](#)

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

[Security Considerations for the SAP Authorization and Trust Management Service \[page 3074\]](#)

### 4.1.6.3.6 Create Role Collections with Predefined Roles

As an application developer, you want to create role collections for immediate use. You want to deliver role collections that administrators can use in the cockpit, and easily assign to users, for example in an onboarding process.

## Prerequisites

- You have the Space Developer role in this subaccount (see the related link).

## Context

You define the role collections in the application security descriptor file (`xs-security.json`). These role collections reference role templates. As soon as you've deployed your application, the cockpit displays the role collections. They contain predefined roles.

## Procedure

1. Deploy an application you want to use for creating security artifacts.
2. Edit the application descriptor file (`xs-security.json`) and add the `role-collections` property.

### Sample Code

```
{  
  "role-templates": [  
    {  
      "name": "Viewer",  
      "description": "View Users",  
      "scope-references": [  
        "$XSAPPNAME.Display"  
      ]  
    },  
    {  
      "name": "Manager",  
      "description": "Maintain Users",  
      "scope-references": [  
        "$XSAPPNAME.Display",  
        "$XSAPPNAME.Update"  
      ]  
    }  
  ],  
  "role-collections": [  
    {  
      "name": "UserManagerRC",  
      "description": "User Manager Role Collection",  
      "role-template-references": [  
        "$XSAPPNAME.Viewer",  
        "$XSAPPNAME.Manager"  
      ]  
    }  
  ]  
}
```

3. Go to the folder where the application security descriptor (`xs-security.json`) file is stored.
4. To deploy the security information, create a service using your `xs-security.json` file.

```
cf create-service xsuaa application <service_name> -c xs-security.json
```

### Example

```
cf create-service xsuaa application rolecoll-serv -c xs-security.json
```

5. (If you do not use a manifest file) Bind your application to the service.

```
cf bind-service <application_name> <service_name>
```

### ❖ Example

```
cf bind-service rcpropertyapp rolecoll-serv
```

You have created a role collection that is visible in the cockpit. It contains predefined roles. Using the cockpit, administrators can assign this role collection to users.

## Related Information

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

[Adding Authentication and Authorization \[page 499\]](#)

[Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#)

[Mapping Role Collections in the Subaccount \[page 2281\]](#)

[Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#)

## 4.1.6.3.7 Validation and Revocation of Access Tokens

To confirm that access tokens were truly issued by the SAP Authorization and Trust Management service and are still valid, applications check the tokens for validity.

Access tokens are JSON web tokens (JWT). When the SAP Authorization and Trust Management service issues access tokens for an application, the service sets a validity and digitally signs the token. The default validity for a token is 12 hours, but you can set the validity as part of the OAuth configuration in the application security descriptor (`xs-security.json`).

Applications must reject the access token in the following cases:

- The token is no longer valid.  
The token could have expired or has been revoked.
- The signature on the token doesn't match.

## Online or Offline Validation

As an application developer, you have the design decision, whether to perform validation of the token online or offline.

In online validation, you send a request to the SAP Authorization and Trust Management service to determine that the access token is still valid and hasn't been revoked. To check the validity, use the `introspect` endpoint of the standard Cloud Foundry API.

Your application can perform offline validation with the help of our client libraries. You must get the public key from the SAP Authorization and Trust Management service and use the key to check the validity. While validating a token offline can ensure that your application can continue to operate if the SAP Authorization and Trust Management service is unavailable, your application continues to accept revoked tokens until they reach the end of their validity. A malicious user with a revoked token still has access as long as the token is still valid.

To revoke a token, use the `revoke` endpoint of the standard Cloud Foundry API.

## Related Information

[Rotate Signing Keys of Access Tokens \[page 2308\]](#)

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

<https://docs.cloudfoundry.org/api/uaa/version/74.4.0/index.html#introspect-token> ↗

<https://docs.cloudfoundry.org/api/uaa/version/74.4.0/index.html#revoke-tokens> ↗

[XSUAA authentication and authorization service integration libraries and samples for authenticating users and services on GitHub](#) ↗

### 4.1.6.3.8 Include Tokens from Corporate Identity Providers or SAP Cloud Identity Services in Tokens of the SAP Authorization and Trust Management Service

In scenarios where your SAP BTP application interacts with applications that don't support tokens issued by the SAP Authorization and Trust Management service, but require a token from a corporate identity provider or SAP Cloud Identity Services, you can configure SAP Authorization and Trust Management service to include this token.

## Prerequisites

- Trust with an SAP Cloud Identity Services tenant is configured with OpenID Connect (OIDC).  
For more information, see [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#).
- Trust with a corporate identity provider by your SAP Cloud Identity Services is configured with OIDC.  
For more information, see [Configure Trust with OpenID Connect Corporate Identity Provider](#) in the SAP Cloud Identity Services documentation.
- The initial logon of your application uses the authorization code flow.

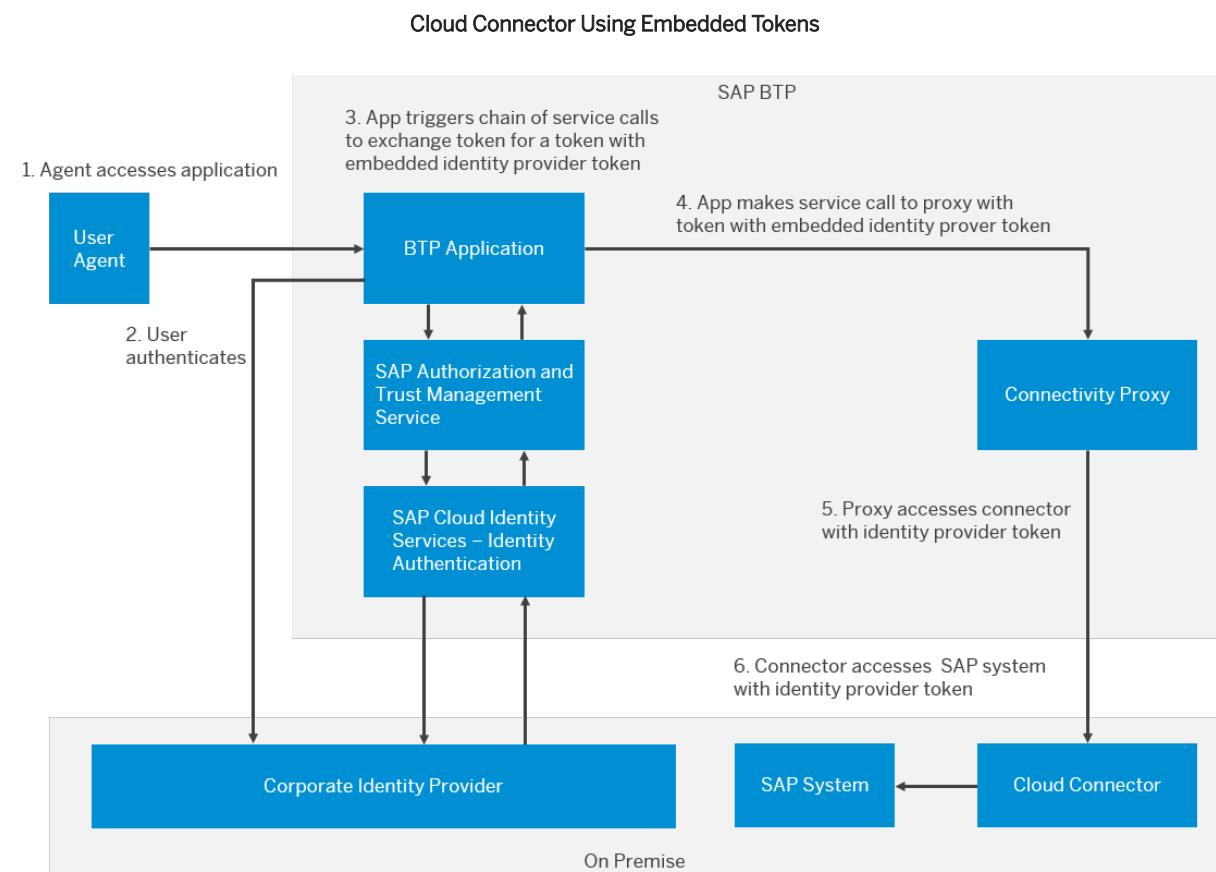
## Context

Imagine you have an on-premise SAP system that trusts your corporate provider. You want to access data in that system from an application on SAP BTP. Your application can get a token from the SAP Authorization and Trust Management service, but you don't want to or can't configure your SAP system to trust that token.

In this scenario, configure the SAP Authorization and Trust Management service to embed the token of a corporate identity provider trusted by the trusted SAP Cloud Identity Services tenant. In this example, before the SAP BTP application makes a service call to the Cloud Connector proxy, the application calls the SAP

Authorization and Trust Management service to exchange its current token for a new token that embeds a token from the identity provider. Cloud Connector presents the identity provider token to the on-premise SAP system.

The following figure illustrates this scenario.



### ⚠ Caution

Token embedding is provided as a best-effort solution. The BTP application must be designed to fail gracefully when the SAP Authorization and Trust Management service is unable to embed tokens from SAP Cloud Identity Services or the corporate identity provider. SAP Authorization and Trust Management service records such events in the audit log.

## Procedure

1. When deploying your application, ensure that the application security descriptor (`xs-security.json`) sets the `system-attributes` with one of the following options:

| Value                   | Description  |
|-------------------------|--|
| ias-corporate-idp-token | Embeds the token of the corporate identity provider trusted by your SAP Cloud Identity Services tenant in the token issued by the service. |
| ias-token               | Embeds the token of your trusted SAP Cloud Identity Services tenant in the token issued by the service.                                    |

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

2. In the SAP Cloud Identity Services application that represents your SAP BTP application, add the self-defined attribute **xsuaa-persist-corporate-idp-token** with the **Expression** source and **true** value.

This attribute ensures that the service knows to embed the tokens.

For more information, see [Configuring Attributes Based on Flexible Expressions](#) in the SAP Cloud Identity Services documentation.

## Results

The service can embed tokens of trusted identity providers in its tokens.

## Next Steps

If you use Cloud Connector, ensure that the service doesn't trust the SAP Authorization and Trust Management service. To use the token of the corporate identity provider, ensure that the service doesn't trust the SAP Cloud Identity Services tenant, too.

For more information, see [Configure Trusted Entities in the Cloud Connector](#) in the documentation of SAP BTP Connectivity.

### 4.1.6.3.9 Best Practices When Handling Access Tokens

Calls to SAP Authorization and Trust Management service can fail for many reasons, including network latency, connection errors, service downtime, idle connections closed by routing components and so on. The following sections provide recommendations on when to retry connections and the caching of tokens.

How your application handles access tokens has an impact on its resiliency. Cache tokens to avoid running into the rate limiting restrictions of the SAP Authorization and Trust Management service. Retry with cached tokens instead of always getting a new one.

For recommended configuration settings, see [Security Considerations for the SAP Authorization and Trust Management Service \[page 3074\]](#).

## Retry Mechanism

When your application or service tries to get a token and fails, retry the connection. A retry mechanism helps your application handle temporary issues. We recommend retrying token retrieval three times by default. The following table lists when to retry and how often.

Retry and Backoff for Different HTTP Responses

| Event Type       | HTTP Status Code   | Retry                         | Backoff Policy                                      |
|------------------|--|-------------------------------|---|
| Connection error | None   | Yes, retry up to three times. | Initial retry after 300 ms with a multiplier of 2.  |
| Timeout          | <ul style="list-style-type: none"><li>• 408: Request timeout</li><li>• 502: Bad gateway</li><li>For example, request timeout from a Cloud Foundry Gorouter.</li><li>• 504: Gateway timeout</li></ul> | Yes, retry up to three times. |   |
| Rate Limiting    | 429: Too many requests   | Yes, retry up to three times. | Base retry on the <code>Retry-After</code> header.  |
| Other Events     | <ul style="list-style-type: none"><li>• 500: Internal server error</li><li>• 503: Service unavailable</li></ul>  | Yes, retry up to three times. | Initial retry after 1000 ms with a multiplier of 3. |
| Access Control   | <ul style="list-style-type: none"><li>• 401: Unauthorized</li><li>• 403: Forbidden</li></ul>   | No, don't retry.              | None.   |

## Token Cache

We recommend that you cache access tokens in your application as long as they're valid, instead of always getting a new token. If you constantly get a new token, you risk running up against the rate limits in the service and downtime for your application.

For more information, see [Rate Limiting \[page 3083\]](#).

However, renewing tokens before they actually expire improves the resiliency of your applications. If your application only renews tokens at their expiration and the service isn't available at that moment, your users experience a downtime. For example, if your token validity is set to 12 hours, try to renew the token after 6 hours.

Having a local cache also improves the performance of your applications.

Implement a method to remove invalid tokens from your cache. When your application calls with a token from your cache and receives the return codes in the following table, remove the token from your cache and request a new token.

## Triggers for Removing Tokens from Your Cache

| HTTP Status Code  | Potential Causes  | Application Response  |
|-------------------|---|---|
| 401: Unauthorized | <ul style="list-style-type: none"><li>The signing key of the token was rotated.<br/>For more information, see <a href="#">Rotate Signing Keys of Access Tokens [page 2308]</a>.</li><li>Token expired and your application hasn't removed it from your cache yet.</li></ul> | Remove the token from the cache and request a new token.  |
| 403: Forbidden    | Occurs if the token is valid but a scope is missing.  | <p><b> ⓘ Note</b></p> <p>Scopes of the client or user can have changed during the caching period.</p> |

### → Tip

SAP Authorization and Trust Management service ensures that keys remain valid for 2 hours after key rotations. Exception: When subaccount administrators force a key rotation, the key is immediately invalid. We recommend that you configure your cache with a 2-hour time-to-live (TTL). If you use a longer TTL, your application must be able to invalidate cache entries when the service responds with 401. Have your application only retry once on failure.

### ⚠ Restriction

When administrators force key rotation, applications that use cached tokens must accept a downtime until their caches have been cleared.

## Related Information

<https://github.com/SAP/cloud-security-services-integration-library/tree/main/token-client#cache-configuration> ↗

## 4.1.6.4 Reference Information

Use additional reference information such as the syntax required to set the properties and values defined in the application security descriptor file.

You can also authenticate Node.js applications. A collection of Node.js packages developed by SAP is provided as part of the Cloud Foundry environment at SAP BTP

### [Application Security Descriptor Configuration Syntax \[page 545\]](#)

The syntax required to set the properties and values defined in the `xs-security.json` application security descriptor file.

### [Authentication for Node.js Applications \[page 563\]](#)

A collection of Node.js packages developed by SAP is provided as part of the Cloud Foundry environment at SAP BTP.

### 4.1.6.4.1 Application Security Descriptor Configuration Syntax

The syntax required to set the properties and values defined in the `xs-security.json` application security descriptor file.

```
{
  "xsappname" : "node-hello-world",
  "scopes"     : [ {
    "name"      : "$XSAPPNAME.Display",
    "description": "display" },
    {
      "name"      : "$XSAPPNAME.Edit",
      "description": "edit" },
    {
      "name"      : "$XSAPPNAME.Delete",
      "description": "delete",
      "granted-apps": [ "$XSAPPNAME(application,business-partner)" ]
    }
  ],
  "attributes" : [ {
    "name"      : "Country",
    "description": "Country",
    "valueType"  : "string" },
    {
      "name"      : "CostCenter",
      "description": "CostCenter",
      "valueType"  : "string" }
  ],
  "role-templates": [ {
    "name"          : "Viewer",
    "description"   : "View all books",
    "default-role-name": "Viewer: Authorized to Read All Books",
    "scope-references": [
      "$XSAPPNAME.Display" ],
    "attribute-references": [
      {
        "name"      : "Country",
        "default-values": [
          "USA", "Germany"
        ]
      }
    ]
  }
]
```

```

        ],
    },
    "name" : "Editor",
    "description" : "Edit, delete books",
    "scope-references" : [
        "$XSAPPNAME.Edit",
        "$XSAPPNAME.Delete" ],
    "attribute-references" : [
        "Country",
        "CostCenter" ]
}
],
"role-collections": [
{
    "name": "UserViewerRC",
    "description": "User Viewer Role Collection",
    "role-template-references": [ "$XSAPPNAME.Viewer" ]
},
{
    "authorities": [ "$ACCEPT_GRANTED_AUTHORITIES" ],
    "oauth2-configuration": {
        "token-validity": 900,
        "redirect-uris": [ "https://myapp.cfapps.eu10.hana.ondemand.com", "https://myapp.mydomain.com/my/logout" ],
        "credential-types": [ "binding-secret", "x509" ]
    },
    "xsenableasyncservice": "true"
}
]
}

```

#### → Tip

Try out the tutorials for the SAP Authorization and Trust Management service to get familiar with using the application security descriptor in the Cloud Foundry environment of SAP BTP.

See [Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#).

## xsappname

Use the `xsappname` property to specify the name of the application that the security description applies to.

```
"xsappname" : "<app-name>" ,
```

### Naming Conventions

Bear in mind the following restrictions regarding the length and content of an application name in the `xs-security.json` file:

- The following characters can be used in an application name of the Cloud Foundry environment at SAP BTP: aA–zz, 0–9, – (hyphen), \_ (underscore), / (forward slash), and \ (backslash).
- The maximum length of an application name is 100 characters.

## tenant-mode (Custom Option)

Use the custom `tenant-mode` property to define the way the tenant's OAuth clients get their client secrets.

During the binding of the xsuaa service, the UAA service broker writes the tenant mode into the credential section. The application router uses the tenant mode information for the implementation of multitenancy with the application service plan.

### ↳ Sample Code

```
{  
  "xsappname"      : "<application_name>",  
  "tenant-mode"    : "shared",  
  "scopes"         : [  
    {  
      "name"          : "$XSAPPNAME.Display",  
      "description"   : "display"  
    }  
  ]  
}
```

### Syntax

```
"tenant-mode" : "shared"
```

The following tenant modes are available:

#### Tenant Modes

| Value     | Description   |
|-----------|---|
| dedicated | Default value. An OAuth client gets a separate client secret for each subaccount.   |
| shared    | An OAuth client always gets the same client secret. It's valid in all subaccounts. The application service plan uses this tenant mode.              |
| external  | A tenant has multiple subscriptions to applications. For each subscription to an application, the tenant gets an OAuth client with a client secret. |

## scopes

In the application security file (`xs-security.json`), the `scopes` property defines an array of one or more security scopes that apply for an application. You can define multiple `scopes`; each scope has a name and a short description. The list of scopes defined in the `xs-security.json` file is used to authorize the OAuth client of the application with the correct set of **local** and **foreign** scopes; that is, the permissions the application requires to be able to respond to all requests.

```
"scopes": [  
  {  
    "name" : "$XSAPPNAME.Display",  
    "description" : "display"  
  },  
  {  
    "name" : "$XSAPPNAME.Edit",  
    "description" : "edit"  
  },  
  {  
    "name" : "$XSAPPNAME.Delete",  
    "description" : "delete",  
    "granted-apps" : [ "$XSAPPNAME(application,business-partner)" ]  
  }  
]
```

```
    }
```

## Local Scopes

All scopes in the `scopes` file are local, that is, application specific. Local scopes are checked by the application's own application router or checked programmatically within the application's runtime container. In the event that an application needs access to other services of the Cloud Foundry environment on behalf of the current user, the provided access token must contain the required foreign scopes. Foreign scopes are not provided by the application itself; they're checked by other sources outside the context of the application.

In the `xs-security.json` file, "local" scopes must be prefixed with the variable `<$XSAPPNAME>` at run time. The variable is replaced with the name of the corresponding local application name.

### → Tip

The variable `<$XSAPPNAME>` is defined in the application's deployment manifest description (`manifest.yml`).

## Foreign Scopes

Usually, "foreign" scopes include the service plan and the name of the application to which the scope belongs. For more information, see [Referencing the Application](#). Use the following syntax:

```
$XSAPPNAME(<service_plan>,<xsappname>).<local_scope_name>
```

### ❖ Example

```
"$XSAPPNAME(application,business-partner).Create"
```

## Granting Scopes to Another Application

If you want to grant a scope from this application to another application for a user scenario, this application needs to grant access to the scope for the application that wants to use this scope. Using the `granted-apps` property in the `scopes` section, you can specify the application you want to grant your scope to. The other application (referenced as `<service_plan>,<foreign_xsappname>`) receives the scope as a "foreign" scope. For more information, see the related link.

Here is the syntax in the security descriptor of the application that grants the scope. For more information, see [Referencing the Application](#).

```
"granted-apps" : [ "$XSAPPNAME(<service_plan>,<foreign_xsappname>)" ]
```

### ❖ Example

```
"granted-apps" : [ "$XSAPPNAME(application,business-partner)" ]
```

If you want to grant a scope from this application to another application for a client credentials scenario, use the `grant-as-authority-to-apps` property in the `scopes` section. In this case, the scopes are granted as **authorities**. Specify the other application by name. For more information, see the related link.

Here is the syntax in the security descriptor of the application that grants the scope:

```
"grant-as-authority-to-apps" : [ "$XSAPPNAME(<service_plan>,<foreign_xsappname>)" ]
```

## • Example

```
"grant-as-authority-to-apps" : [ "$XSAPPNAME(application,business-partner)" ]
```

## Naming Conventions

Bear in mind the following restrictions regarding the length and content of a scope name:

- The following characters can be used in a scope name: aA–zz, 0–9, – (hyphen), \_ (underscore), / (forward slash), \ (backslash), and : (colon).
- Scope names can't start with a leading . (period). For example, .myScopeName1.
- The maximum length of a scope name, including the fully qualified application name, is 193 characters.

For other limits when using the properties, see [Limits for the Application Security Descriptor \[page 3088\]](#).

## attributes

In the application security descriptor (`xs-security.json`), the `attributes` property enables you to define an array, listing one or more attributes that are applied to the role templates also defined in the `xs-security.json` element. This element is only relevant for user scenarios. You can define multiple attributes.

```
"attributes" : [
  {
    "name" : "Country",
    "description" : "Country",
    "valueType" : "s" },
  {
    "name" : "CostCenter",
    "description" : "CostCenter",
    "valueType" : "string" }
],
```

The `attributes` element is only relevant for a user scenario. Each element of the attributes array defines an attribute. These attributes can be referenced by role templates. There are multiple sources of attributes:

- Static attributes  
Use the SAP BTP cockpit to assign the value of the attributes. You can use the static value.
- Attributes from an identity provider  
You can reference the attributes provided by an identity provider. The token is issued by the configured identity provider during authentication. You find the attribute value in the configuration of your identity provider. The attributes provided by the identity provider appear as an attribute in the JSON web token, if the user has assigned the corresponding roles. You can use the attributes to achieve instance-based authorization checks.
- Unrestricted attributes  
In this case, you want to express that it's not necessary to set a specific value for this attribute. The behavior is the same as if the attribute would not exist for this role.

For more information, see the related link about attributes.

The `attributes` definition can take the following properties:

#### attributes Properties

| Key                        | Description   | Example              |
|----------------------------|---|----------------------|
| <code>name</code>          | The name of the attribute with a value to apply when building the role template   | <code>Country</code> |
| <code>description</code>   | A short summary of the attribute defined  | <code>Country</code> |
| <code>valueType</code>     | The type of value expected for the defined attribute; possible values are: <code>string</code> (or <code>s</code> ), <code>int</code> (integer), or <code>date</code>   | <code>int</code>     |
| <code>valueRequired</code> | By default, every attribute needs dedicated attribute values. The default value is <code>true</code> .<br><br>For more information, see <a href="#">Relationship Between default-values of attribute-references and valueRequired</a> . | <code>true</code>    |

#### Naming Conventions

Bear in mind the following restrictions regarding the length and content of attribute names in the `xs-security.json` file:

- The following characters can be used to declare an xs-security attribute name in the Cloud Foundry environment: “aA”-“zZ”, “0”-“9”, “\_” (underscore)
- The maximum length of a security attribute name is 64 characters.

For other limits when using the properties, see [Limits for the Application Security Descriptor \[page 3088\]](#).

## role-templates

In the application-security file (`xs-security.json`), the `role-templates` property enables you to define an array listing one or more roles (with corresponding scopes and any required attributes), which are required to access a particular application module. You can define multiple `role-templates`, each with its own scopes and attributes.

Role templates can be delivered with default values for each attribute reference. When you deploy role templates with default values for each attribute reference, you create default roles.

`attribute-references` can contain a JSON array of multiple objects.

```
"role-templates": [
  {
    "name" : "Viewer",
    "description" : "View all books",
    "default-role-name" : "Viewer: Authorized to Read All Books",
    "scope-references" : [
      "$XSAPPNAME.Display"
    ],
    "attribute-references": [
      ...
    ]
  }
]
```

```
{
  "name" : "Country",
  "default-values" : [
    "USA", "Germany"
  ]
}
```

`attribute-references` can contain a JSON array of string.

A role template must be instantiated. This is especially true with regard to any attributes defined in the role template and the specific attribute values, which are subject to customization and, as a result, can't be provided automatically. Role templates that only contain "role-templates": [ { "name": "Viewer", "description": "View all books", "default-role-name": "Viewer: Authorized to Read All Books", "scope-references": [ "\$XSAPPNAME.Display" ], "attribute-references": [ "Country" ] }, ] "local""role-templates": [ scopes can be instantiated without user interaction. The same is also true for "foreign"scopes where the scope **owner** has declared consent in a kind of allowlist (for example, either for "public" use or for known "friends").

### ⓘ Note

The resulting (application-specific) role instances need to be assigned to the appropriate user groups.

role-template Properties

| Key               | Description  | Example                              |
|-------------------|--|--------------------------------------|
| name              | The name of the role to build from the role template   | Viewer                               |
| description       | A short summary of the role to build   | View all books                       |
| default-role-name | The name of the role in Unicode<br>The name can be up to 255 characters long. The naming conventions of role-templates don't apply here. | Viewer: Authorized to Read All Books |
| scope-references  | The security <b>scope</b> to apply to the application-related role   | \$XSAPPNAME.Display                  |

| Key                  | Description  | Example  |
|----------------------|--|--|
| attribute-references | <p>One or more attributes to apply to the built role. You can use a JSON array of objects or of string.</p> <p>If you use an array of objects, <code>name</code> is the name of the attribute. Use the attribute name specified in the <code>attributes</code> section.</p> <p><code>default-values</code>, for example <code>"default-values": [ "LeaveRequestWorkflow" ]</code>. If you don't want to configure default values, don't specify the <code>default-values</code> element.</p> <p>For more information, see <a href="#">Relationship between default-values of attribute-references and valueRequired</a>.</p> | <p>↳ <b>Sample Code</b></p> <pre>{   "name" : "Country",   "default-   values" : [ "USA",   "Germany" ] }</pre> <p>(an array of objects)</p> <p>[ "Country" ] (an array of string)</p> |

#### → Tip

If you want to deliver attributes that aren't restricted by attribute values, set `"valueRequired": false`.

## Naming Conventions

Bear in mind the following restrictions regarding the length and content of role-template names in the `xs-security.json` file:

- The following characters can be used to declare an xs-security role-template name: “aA”-“zZ”, “0”-“9”, “.” (period), “-” (hyphen), “\_” (underscore).
- The maximum length of a role-template name is 64 characters.

For other limits when using the properties, see [Limits for the Application Security Descriptor \[page 3088\]](#).

## role-collections

The optional `role-collections` property enables you to define role collections with predefined roles. Administrators use these predefined role collections. They can assign them to users during the initial setup of SAP BTP.

The `role-collections` property only makes sense if application developers reference role templates that can create default roles at deployment time.

#### ⓘ Note

The `role-collections` element can only reference role templates of the same subaccount.

## ↔ Sample Code

```
"role-collections": [
  {
    "name": "Employee",
    "description": "Employee roles",
    "role-template-references": [
      "$XSAPPNAME.Employee"
    ]
  }
]
```

role-collections Properties

| Key                      | Description  | Example   |
|--------------------------|--|---|
| name                     | The name of the role collection to build   | Employee  |
|                          | <b>⚠ Caution</b><br><br>If you change the name of a role collection and update the service instance, you delete the role collection with the old name and create a role collection with the new name. This means that the newly created role collection doesn't have any user assignments. |   |
| description              | A short summary of the role collection to build  | Employee roles  |
| role-template-references | The role templates referenced by the <code>role collections</code> property  | \$XSAPPNAME.Employee<br>Syn-<br>tax\$XSAPPNAME(<service_plan>, <xsappname>) |

## Naming Conventions

Bear in mind the following restrictions regarding the length and content of `role-collections` names in the `xs-security.json` file:

- The maximum length of a role-collection name is 64 characters.
- The maximum length of a role-collection description is 1000 characters.
- The `role-template-references` (mandatory) element is an array with references to role template definitions from the Cloud Foundry application or from other Cloud Foundry applications.

## → Tip

We recommend that you reference all role templates using either the \$XSAPPNAME or \$XSAPPNAME( <service\_plan>, <XSAPPNAME>) prefix to link to the correct application where the role template is defined (see [Referencing the Application](#)).

## Conditions

The role templates must fulfill one of the following conditions regarding attributes:

- Either they don't have attribute references.
- Or default attribute values are defined. It's also possible to add the "valueRequired" : false property to an attribute.

This makes sure that a default role is automatically created for the role-templates after developers have deployed the application. The role collections use the predefined roles, which have been created automatically.

For other limits when using the properties, see [Limits for the Application Security Descriptor \[page 3088\]](#).

## authorities

To enable one (sending) application to accept and use the authorization scopes granted by another application, each application must be bound to its own instance of the xsuaa service. This is required to ensure that the applications are registered as OAuth 2.0 clients at the UAA. You must also add an authorities property to the security descriptor file of the sending application that is requesting an authorization scope. In the authorities property of the sending application's security descriptor, you can either specify that **all** scope authorities configured as grantable in the receiving application should be accepted by the application requesting the scopes, or alternatively, only individual, named, scope authorities:

- Request and accept **all** authorities flagged as "grantable" in the receiving application's security descriptor:

### ↔ Sample Code

Specifying Scope Authorities in the Sending Application's Security Descriptor

```
"authorities": [ "$ACCEPT_GRANTED_AUTHORITIES" ]
```

- Request and accept only the "specified" scope authority that is flagged as grantable in the specified receiving application's security descriptor. For more information, see [Referencing the Application](#).

### ↔ Sample Code

Specifying Scope Authorities in the Sending Application's Security Descriptor

```
"authorities": [ "<ReceivingApp>.ForeignCall",
"<ReceivingApp2>.ForeignCall", ]
```

Since both the sending application and the receiving application are bound to UAA services using the information specified in the respective application's security descriptor, the sending application can accept and use the specified authority from the receiving application. The sending application is now authorized to access the receiving application and perform some tasks.

### Note

The granted authority always includes the prefixed name of the application that granted it. This information tells the sending application, which receiving application has granted which set of authorities.

The scope authorities listed in the sending application's security descriptor must be specified as "grantable" in the receiving application's security descriptor.

### Tip

To flag a scope authority as "grantable", add the `grant-as-authority-to-apps` property to the respective scope in the receiving application's security descriptor.

For more information, see the related link.

## **oauth2-configuration (Custom Option)**

Use the custom `oauth2-configuration` property to define custom values for the OAuth 2.0 clients, such as the token validity and redirect URIs.

The `xsuaa` service broker registers and uses these values for the configuration of the OAuth 2.0 clients.

### Sample Code

```
"oauth2-configuration": {  
    "token-validity": 43200,  
    "redirect-uris": [  
        "https://myapp.cfapps.eu10-004.hana.ondemand.com",  
        "https://myapp.mydomain.com/my/content"],  
    "refresh-token-validity": 1800,  
    "credential-types": ["binding-secret", "x509"],  
    "system-attributes": ["groups", "rolecollections"],  
    "allowedproviders": ["origin_key1", "origin_key2"]  
}
```

The following configuration keys are available:

#### oauth2-configuration Properties

| Key                    | Description  | Example           |
|------------------------|--|-------------------|
| token-validity         | <p>Sets the token lifetime in seconds for access tokens issued by SAP Authorization and Trust Management service. The value ranges from 60 seconds to 86400 seconds, in other words, from 1 minute to 24 hours.</p> <p>Default: 43200 seconds (12 hours)</p> <div style="background-color: #f0f8ff; padding: 10px; border-radius: 5px;"><p><b>→ Recommendation</b></p><p>Relaxing the token policy means that users reauthenticate less. However, increasing the token validity also means that if a malicious user manages to steal a token, that malicious user has access until the token expires. Keep token validity in the subaccount as short as possible, but not less than 30 minutes.</p></div> <div style="background-color: #f0f8ff; padding: 10px; border-radius: 5px;"><p><b> ⓘ Note</b></p><p>This change applies to applications, which consume or subscribe to this instance. These values override the values set for the subaccount.</p><p>For more information, see <a href="#">Configure Token Policy for SAP Authorization and Trust Management Service [page 2316]</a>.</p></div> | 1800 (30 minutes) |
| refresh-token-validity | <p>Sets the token lifetime in seconds for refresh tokens issued by SAP Authorization and Trust Management service. The value ranges from 60 seconds to 31536000 seconds, in other words, from 1 minute to 1 year.</p> <p>Default: 604800 seconds (7 days)</p>  | 604800 (7 days)   |

| Key              | Description   | Example   |
|------------------|---|---|
| redirect-uris    | <p>This key contains a list of the redirect URIs that SAP BTP checks for when redirecting. If your landscape domain or custom domain aren't on this list, including wildcards, the Authorization and Trust Management Service won't redirect users there.</p> <p>We support explicit wildcards, namely domain relaxing and arbitrary paths. For example: "https://*.mydomain.com/callback/**"</p> | <pre>[ "https://&lt;application_hostname1&gt;.&lt;landscape_domain&gt;&lt;path&gt;" , "https://&lt;application_hostname2&gt;.&lt;custom_domain&gt;&lt;path&gt;" ]</pre> |
|                  | <p>For more information, see <a href="#">Listing Allowed Redirect URIs [page 3075]</a>.</p>   |   |
| credential-types | <p>Specifies the types of secrets available for binding applications to the service instance. The first type listed is the default type of secret. Otherwise, specify the type of secret when you create the binding or service key.</p>  | <pre>[ "binding-secret" ]</pre>   |
|                  | <p>For more information, see <a href="#">Service Instance Secrets [page 2294]</a>.</p>  |   |
|                  | <p>The default value is <code>instance-secret</code>. With <code>instance-secret</code> all bindings of a service instance of the SAP Authorization and Trust Management service share a single instance secret.</p>  |   |
|                  | <p>→ <b>Recommendation</b></p> <p>Use <code>binding-secret</code> or <code>x509</code> so you can rotate the secret of a single binding without affecting other bindings of the service instance.</p>   |   |
|                  | <p>For more information, see <a href="#">Rotating Secrets [page 3076]</a>.</p>  |   |

| Key               | Description   | Example                          |
|-------------------|---|----------------------------------|
| system-attributes | <p>Includes the system attributes in the JWT. If you don't define a value, the system includes the attributes for groups and role collections by default. If the size of the JWT becomes an issue for you, you can explicitly remove them. For example: "system-attributes": [ ],</p> <p>You can also have the service include a refresh token of a trusted identity provider.</p> <p>For more information, see <a href="#">Include Tokens from Corporate Identity Providers or SAP Cloud Identity Services in Tokens of the SAP Authorization and Trust Management Service [page 540]</a>.</p> <p><b>Values:</b></p> <ul style="list-style-type: none"> <li>• <b>"groups"</b> includes the <code>xs.saml.groups</code> attribute.</li> <li>• <b>"rolecollections"</b> includes the <code>xs-rolecollections</code> attribute.</li> <li>• <b>"ias-corporate-idp-token"</b> includes the token of the corporate identity provider trusted by your SAP Cloud Identity Services tenant.</li> <li>• <b>"ias-token"</b> includes the token of the trusted SAP Cloud Identity Services tenant.</li> </ul> | [ "groups" , "rolecollections" ] |

| Key              | Description  | Example                          |
|------------------|--|----------------------------------|
| allowedproviders | <p>Includes an array of allowed identity providers. The default is that all identity providers are allowed.</p> <p>As an option, developers can configure on application level which identity providers can be used for business users to log on to a certain application. The value of <code>allowedproviders</code> is the origin key of your identity provider. You find the origin key in your subaccount in the ► <i>Security</i> ► <i>Trust Configuration</i> page of your identity providers.</p> | [ "origin_key1", "origin_key2" ] |

#### → Recommendation

We recommend that you configure allowed identity providers on subaccount level. For more information, see [Using Multiple Identity Providers from the Same Subaccount \[page 2233\]](#).

## xsenableasyncservice (Custom Option)

The `xsenableasyncservice` element controls whether the `cf create-service` and `cf update-service` commands of the Cloud Foundry command line interface (cf CLI) are executed synchronously (false) or asynchronously (true).

#### → Remember

The Cloud Controller of Cloud Foundry times out after 60 seconds for the synchronous execution of the commands. For this reason, we recommend that you use asynchronous execution for multitenant applications or reuse services.

#### ⓘ Note

You usually need to wait for the completion of `cf create-service` and `cf update-service` even if you use asynchronous mode. Otherwise you might run into conflicts due to parallel update operations on the OAuth 2.0 clients.

#### ↔ Sample Code

```
"xsenableasyncservice": "true"
```

## Syntax

```
"xsenableasyncservice": "false"
```

| Value           | Description  |
|-----------------|--|
| false (default) | Cloud Foundry command line interface commands are executed synchronously.  |
| true            | Cloud Foundry command line interface commands are executed asynchronously. |

## Referencing the Application

If you want to grant scopes to an application for example, you must reference this application. Depending on where the application is located, you can reference an application in multiple ways:

- Referencing the application of the current `xs-security.json` file
- Referencing a foreign application that is located in the same subaccount

### Application References

| Description  | Syntax   |
|--|--|
| Application in the current <code>xs-security.json</code> file      | <code>\$XSAPPNAME</code>   |
| Application in the same subaccount                                 | <code>\$XSAPPNAME(&lt;service_plan&gt;, &lt;xsappname&gt;)</code>                                |
| Reference to any service instance in the same subaccount and space | <div><p><b>① Note</b></p><p>Currently, you can only use the application service plan.</p></div>  |
|  | <div><p><b>❖ Example</b></p><p><code>\$XSAPPNAME(application, business-partner)</code></p></div> |
| Reference to any service instance in the same subaccount and space | <code>\$XSSERVICENAME(&lt;service_instance_name&gt;)</code>                                      |
|  | This is the service instance name you used when you created it.                                  |

You can use these references with the following properties:

#### Properties

| Property                   | Description  | Example  |
|----------------------------|--|--|
| granted-apps               | Granting scopes to other applications.   | <p>↳ Sample Code</p> <pre>"granted-apps" :<br/>[ "\$XSAPPNAME(application,business-partner)" ]</pre><br><pre>"granted-apps" :<br/>[ "\$XSAPPNAME(application,business-partner1)", "\$XSAPPNAME(application,business-partner2)" ]</pre>                             |
| grant-as-authority-to-apps | Use this property if you want to grant a scope to other applications for a client credential scenario. | <p>↳ Sample Code</p> <pre>"grant-as-authority-to-apps" :<br/>[ "\$XSAPPNAME(application,business-partner)" ]</pre><br><pre>"grant-as-authority-to-apps" :<br/>[ "\$XSAPPNAME(application,business-partner1)", "\$XSAPPNAME(application,business-partner2)" ]</pre> |
| authorities                | Granting authorities (for a client credentials scenario)   | <p>↳ Sample Code</p> <pre>"authorities":<br/>[ "\$XSAPPNAME(application,business-partner)" ]</pre>   |
| scope-references           | Referencing scopes in role templates   | <p>↳ Sample Code</p> <pre>\$XSAPPNAME.Display</pre>  |
| foreign-scope-references   | Using this property, you can reference scopes in foreign applications (for a user scenario).           | <p>↳ Sample Code</p> <pre>"foreign-scope-references":<br/>[ "\$XSAPPNAME(application,business-partner)" ]</pre><br><pre>"foreign-scope-references":<br/>[ "\$XSAPPNAME(application,business-partner1)", "\$XSAPPNAME(application,business-partner2)" ]</pre>       |

## Relationship Between default-values of attribute-references and valueRequired

When you define role templates and attributes in the `xs-security.json` file, you need to know that there's a relationship between `default-values` of `attribute-references` in the role template and `valueRequired` in the attribute. What is important to know is that you configure `valueRequired` in the current attribute definition independently from any role template. The attributes are later referenced in the role templates using the `attribute-references` element.

This means that an attribute can be referenced in multiple role templates. You can set `default-values` for the first role template and not for the second one. If you define an attribute with `"valueRequired": true` (see rows 1 and 3), a default role can be generated automatically for the first role template only, but not for the second one.

### Configuration Examples

| Use Case   | Role Template  | Attribute  |
|--|--|--|
| A default role with default values is generated automatically. Administrators must set attribute values for their roles.   | <pre>"default-values":<br/>[ "&lt;attribute_value&gt;" ]</pre> | <code>"valueRequired" : true</code><br>or no <code>valueRequired</code> key at all |
| A default role with default values is generated automatically. Administrators don't have to set attribute values for their role. A role that isn't restricted by attributes (unrestricted) is created. | <pre>"default-values":<br/>[ "&lt;attribute_value&gt;" ]</pre> | <code>"valueRequired" : false</code>   |
| A default role is generated automatically. Administrators don't have to set attribute values for their roles. A role that isn't restricted by attributes (unrestricted) is created.                    | No <code>default-values</code> element                         | <code>"valueRequired" : false</code>   |

### ⚠ Restriction

When you define role templates and attributes, observe the following configuration restriction:

- Do not add `attribute-references` without `default-values`, where `attribute` is defined with `"valueRequired" : true`.

When you update role templates and attributes, observe the following configuration restrictions:

- Do not change `default-values` of `attribute-references`.
- Do not change the `"valueRequired"` property from `false` to `true`.

## Related Information

[Attributes \[page 2288\]](#)

Tutorials for the SAP Authorization and Trust Management Service [page 527]

Security Considerations for the SAP Authorization and Trust Management Service [page 3074]

Configuration Options for the SAP Authorization and Trust Management Service [page 3078]

## 4.1.6.4.2 Authentication for Node.js Applications

A collection of Node.js packages developed by SAP is provided as part of the Cloud Foundry environment at SAP BTP.

SAP BTP includes a selection of standard Node.js packages, which are available for download and use from the SAP NPM public registry to customers and partners. SAP BTP only includes the Node.js packages with long-time support (LTS). For more information, see <https://nodejs.org>. Enabling access to an NPM registry requires configuration using the SAP NPM Registry.

- Besides the standard NPM Package Manager, you can use the NPM Package Manager to download the packages from the npm repository.  
<https://registry.npmjs.org>
- As an alternative option, you can use the SAP CLIENT LIB 1.0. Search for the software component named SAP CLIENT LIB 1.0 on the ONE Support Launchpad.  
[ONE Support Launchpad](#)

The SAP CLIENT LIB 1.0 package contains the following modules:

→ Tip

For more details of the package contents, see the README file in the corresponding package.

Contents of SAP CLIENT LIB 1.0

| Package Name                   | Description   |
|--------------------------------|---|
| <a href="#">@sap/approuter</a> | The application router is the single entry point for the (business) application.          |
| <a href="#">@sap/xssec</a>     | The client security library, including the XS advanced container security API for Node.js |

### @sap/approuter

The application router is the single entry point for the (business) application. It has the responsibility to serve static content, authenticate users, rewrite URLs, and proxy requests to other micro services while propagating user information.

For more information, see the applications section of SAP BTP.

## @sap/xssec

The client security library includes the container security API for Node.js.

Authentication for node applications relies on the usage of the OAuth 2.0 protocol, which is based on central authentication at the User Account and Authentication (UAA) server that then vouches for the authenticated user's identity by means of a so-called OAuth access token. The implementation uses as access token a JSON web token (JWT), which is a signed text-based token formatted according to the JSON syntax.

The trust for the offline validation is created by binding the UAA service instance to your application. The key for validation of tokens is included in the credentials section in the environment variable `<VCAP_SERVICES>`. By default, the offline validation check only accepts tokens intended for the same OAuth 2.0 client in the same UAAsubaccount. This makes sense and covers the majority of use cases. However, if an application needs to consume tokens that were issued either for different OAuth 2.0 clients or for different subaccounts, you can specify a dedicated access control list (ACL) entry in an environment variable named `<SAP_JWT_TRUST_ACL>`. The name of the OAuth 2.0 client has the prefix `sb-`. The content is a JSON string. It contains an array of subaccounts and OAuth 2.0 clients. To establish trust with other OAuth 2.0 clients and/or subaccounts, specify the relevant OAuth 2.0 client IDs and subaccounts.

### ⚠ Caution

For testing purposes, use an asterisk (\*). This setting should never be used for productive applications.

Subaccounts are not used for on-premise systems. The value for the subaccount is `uaa`.

```
SAP_JWT_TRUST_ACL: [ { "clientid": "<OAuth_2.0_client_ID>" , "subaccount" : "<subaccount>" } , . . . ]
```

### ⚠ Caution

The client libraries (`java-security`, `spring-xsuaa`, and `container security api for node.js >= 3.0.6`) have been updated. When using these libraries, setting the parameter `SAP_JWT_TRUST_ACL` has become obsolete. This update comes with a change regarding scopes. For a business application A that wants to call an application B, it's now mandatory that application B grants at least one scope to the calling business application A. Furthermore business application A has to accept these granted scopes or authorities as part of the application security descriptor. You can grant scopes with the `xs-security.json` file. For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#), specifically the sections "Referencing the Application" and "Authorities".

In a typical deployment scenario, your node application consists of several parts, which appear as separate application modules in your manifest file, for example:

- Application logic  
This application module (`<myAppName>/js/`) contains the application logic: code written in Node.js. This module can make use of this *XS Advanced Container Security API for Node.js*.
- UI client  
This application module (`<myAppName>/web/`) is responsible for the UI layer; this module can make use of the application router functionality (defined in the file `xs-app.json`).

### ⓘ Note

The application logic written in Node.js and the application router should be bound to one and the same UAA service instance so that these two parts use the same OAuth client credentials.

To use the capabilities of the container security API, add the module “@sap/xssec” to the dependencies section of your application’s package.json file.

### Note

To enable tracing, set the environment variable DEBUG as follows: DEBUG=xssec: \*.

### Usage

All SAP modules, for example @sap/xssec, are located in the namespace of the SAP NPM registry. For this reason, you must use the SAP NPM registry and the default NPM registry.

Path to the NPM registries:

| NPM Registry         | Path   |
|----------------------|--|
| SAP NPM registry     | @sap:registry = "https://registry.npmjs.org" |
| Default NPM registry | registry = "http://registry.npmjs.org/"      |

If you use express and passport, you can easily plug a ready-made authentication strategy.

### Sample Code

```
var express = require('express');
var passport = require('passport');
var JWTStrategy = require('@sap/xssec').JWTStrategy;
var xsenv = require('@sap/xsenv');
...
var app = express();
...
passport.use(new JWTStrategy(xsenv.getServices({uaa:{tag:'xsuaa'}}).uaa));
app.use(passport.initialize());
app.use(passport.authenticate('JWT', { session: false }));
```

We recommend that you disable the session as in the example above. Each request comes with a JWT token so it is authenticated explicitly and identifies the user. If you still need the session, you can enable it but then you should also implement user serialization/deserialization and some sort of session persistency.

### Container Security API

#### Tip

For more details of the package contents, see the README file in the corresponding package.

## Container Security API

| API                   | Description   |
|-----------------------|---|
| createSecurityContext | <p>Creates the "security context" by validating the received access token against credentials put into the application's environment via the UAA service binding</p> <p>Returns a structure with the following properties:</p> <ul style="list-style-type: none"> <li>• <code>getLogonName</code></li> <li>• <code>getGivenName</code></li> <li>• <code>getFamilyName</code></li> <li>• <code>getEmail</code></li> <li>• <code>getHdbToken</code></li> <li>• <code>getAdditionalAuthAttribute</code></li> <li>• <code>getExpirationDate</code></li> <li>• <code>getGrantType</code></li> </ul>  |
| checkLocalScope       | Checks a scope that is published by the current application in the <code>xs-security.json</code> file   |
| checkScope            | Checks a scope that is published by an application  |
| getToken              | <p>Returns a token that can be used to connect to the SAP HANA database. If the token that the security context has been instantiated with is a foreign token (meaning that the OAuth client contained in the token and the OAuth client of the current application do not match), "null" is returned instead of a token; the following attributes are available:</p> <ul style="list-style-type: none"> <li>• <code>namespace</code><br/>Tokens can be used in different contexts, for example, to access the SAP HANA database, to access another XS advanced-based service such as the Job Scheduler, or even to access other applications or containers. To differentiate between these use cases, the namespace is used. In <code>lib/constants.js</code> we define supported name spaces (for example, <code>SYSTEM</code>).</li> <li>• <code>name</code><br/>The name is used to differentiate between tokens in a given namespace, for example, "HDB" for the SAP HANA database. These names are also defined in the file <code>lib/constants.js</code>.</li> </ul> |
| hasAttributes         | Returns "true" if the token contains any XS advanced user attributes; otherwise "false".  |

| API             | Description   |
|-----------------|---|
| getAttribute    | Returns the attribute exactly as it is contained in the access token. If no attribute with the given name is contained in the access token, "null" is returned. If the token that the security context has been instantiated with is a foreign token (meaning that the OAuth client contained in the token and the OAuth client of the current application do not match), "null" is returned regardless of whether the requested attribute is contained in the token or not. The following attributes are available: <ul style="list-style-type: none"> <li>• name</li> </ul> The name of the attribute that is requested |
| isInForeignMode | Returns "true" if the token, that the security context has been instantiated with, is a foreign token that was not originally issued for the current application, otherwise "false".  |
| getIdentityZone | Returns the subaccount that the access token has been issued for.   |

## 4.1.7 Setting Up Database Artifacts

Learn how to set up database artifacts using SAP HANA.

SAP HANA is an innovative in-memory relational database management system that you can leverage when developing in the SAP BTP, Cloud Foundry environment. It minimizes data movements by making it possible for you to work with the data directly in memory.

With SAP HANA, you can leverage current hardware capabilities to increase performance and reduce costs, for example by building applications that integrate the business control logic and database layer.

### ⓘ Note

In the Cloud Foundry environment, the SAP HANA database supports up to 1,000 simultaneous connections per database.

We recommend using SAP HANA Cloud when following a multitarget application (MTA) approach.

To find out more about the tools, data modeling strategy, and development scenarios, see [Developing SAP HANA in the Cloud Foundry Environment \[page 567\]](#).

### 4.1.7.1 Developing SAP HANA in the Cloud Foundry Environment

Find here selected information for SAP HANA database development and references to more detailed sources.

This section gives you information about database development and its surrounding tasks especially if you want to explore the SAP BTP, Cloud Foundry environment. To get more into detail, we have references to other

guides and the SAP HANA Cloud service documentation. The context we're looking at is multitarget application (MTA) development, whereby SAP HANA is the database module and you develop all artifacts in that module.

There are multiple scenarios when you start using SAP HANA Cloud in the SAP BTP, Cloud Foundry environment. For each of these scenarios, we give you a recommendation:

| Scenario   | Recommendation   |
|--|--|
| You're starting a new project and want to leverage SAP HANA capabilities in the cloud.   | We recommend using the SAP Cloud Application Programming Model and SAP HANA Cloud.<br><br>See <a href="#">SAP HANA Cloud [page 568]</a>  |
| You're using SAP HANA service for SAP BTP and SAP HANA Platform 2.0 (XS advanced) on-premise.  | Get to know the differences. Read about features that have been supported by other SAP HANA versions but aren't supported by SAP HANA Cloud: <a href="#">SAP HANA Cloud Compatibility Reference</a>  |
| You're using SAP HANA Platform 2.0 (XS advanced) on-premise.   | SAP HANA extended application services, advanced model, (XS advanced) as the runtime environment for SAP HANA Platform was built similar to the Cloud Foundry environment. It has been developed further and therefore contains features that aren't available in Cloud Foundry. Some features aren't supported from the SAP HANA versions in the cloud. So, there's always some effort, at least when you reach a decent amount of complexity, to make an XS advanced application run in the cloud.<br><br>The new SAP HANA Cloud service made significant moves to reduce footprint and be cloud-ready. Therefore, previously deprecated features have been removed. Read about <a href="#">SAP HANA Cloud Compatibility and Migration Information</a> : <ul style="list-style-type: none"><li>• <a href="#">High Level Feature Compatibility</a></li><li>• <a href="#">Design-time Content Compatibility</a></li></ul><br>You can find another collection of these features and elements in the following SAP Note: <a href="#">2868742</a> |
| You're using SAP HANA Platform 1.0 (XS classic) on-premise. This scenario also includes any usage of XS classic artifacts even if you use XS advanced (compatibility) or the Neo environment | This scenario is presumably a bigger project. The task isn't only to use a different version of SAP HANA but also to make architectural changes: from monolithic applications to microservice-based applications. To give you an overview on things you need to consider, have a look at our guide: <a href="#">Migrating from the Neo Environment to the Multi-Cloud Foundation for SAP BTP (Cloud Foundry and Kyma)</a>  |

## 4.1.7.1.1 SAP HANA Cloud

This section is a quick introduction to the latest SAP HANA offering on SAP BTP, Cloud Foundry environment. In essence, you get to know the most important tools and information about data modeling. This content is meant to provide an overview. Please follow the links to the content you're interested in and start your learning journey.

## Tools

There are two tools you need to know, SAP Business Application Studio and SAP HANA Cockpit.

- SAP Business Application Studio

SAP Business Application Studio is the recommended tool to develop your SAP HANA database artifacts. You can use an [SAP HANA Native Application](#) dev space, which is preconfigured to support the creation of database artifacts.

For SAP Cloud Application Programming Model projects, we recommend to create a [Full Stack Cloud Application](#) dev space and add the following extensions:

- SAP HANA Calculation View Editor
- SAP HANA Tools
- SAP HANA Smart Data Integration Tools

With that configuration you have the editors and tools for the programming model and SAP HANA database artifacts.

- SAP HANA Cockpit

The SAP HANA Cockpit is used to monitor and administer your SAP HANA instances, see [Open the SAP HANA Cockpit](#)

To learn how to access the SAP HANA Cockpit, see [Accessing SAP HANA Cockpit for SAP HANA Cloud](#)

## Data Modeling

To develop your data model, we recommend the SAP Cloud Application Programming Model.

- The SAP Cloud Application Programming Model (CAP) provides a guide explaining the concepts and best practices of domain modeling using CAP CDS, see [Domain Modeling](#).
- To define your data models, queries and expressions, there's a CAP CDS reference guide. It contains samples and best practices that support your learning journey. See [Core Data Services \(CDS\) - Language Reference Documentation for the application programming model](#).
- You have existing database objects and want to use them with CAP CDS. Learn about facade entities in CAP CDS and how to design them. See [Native SAP HANA](#).

When you've created your data model with the SAP Cloud Application Programming Model, you want to leverage more of the SAP HANA power. The SAP HANA Cloud documentation provides an extensive library of guides. Find here selected links that make an easy start.

- [SAP HANA Cloud Modeling Guide for SAP Web IDE Full-Stack](#)
- [SAP HANA Cloud product page, development overview](#)

## References

- [SAP HANA Cloud product page on the Help Portal](#)
- [Feature Scope Description for SAP HANA Cloud](#)

## 4.1.8 Deploying to the Cloud Foundry Environment

Get an overview of available deployment options.

When deploying applications and services to the cloud, you can choose from a number of different approaches and tools, depending on your development setup.

## Multitarget Applications

If you've decided to develop your application or service with the multitarget approach, you can deploy it to SAP BTP using a single `.mtar` archive file. For more information, see [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#).

## Continuous Integration and Delivery

Continuous Integration and Delivery (CI/CD) are two interrelated concepts that can enhance your development process by adding frequent integration, automated testing, and faster delivery. For more information, see [Continuous Integration and Delivery \[page 698\]](#).

## Cloud Foundry Command Line Interface

If you prefer to develop on your local machine, you can deploy your application from the folder on your computer directly to SAP BTP using the Cloud Foundry Command Line Interface (CF CLI). For more information, see [Deploy Business Applications in the Cloud Foundry Environment \[page 702\]](#).

## Docker Images

If you want a higher degree of freedom when realizing your project, you can also deploy your applications and services using Docker images and the Cloud Foundry Command Line Interface (CF CLI). Note that Docker images can't be used in combination with buildpacks. For more information, see [Deploy Docker Images in the Cloud Foundry Environment \[page 703\]](#).

### 4.1.8.1 Multitarget Applications in the Cloud Foundry Environment

A Multitarget application (MTA) is essentially a single application that consists of multiple parts. These parts are created using various technologies and share the same lifecycle.

The MTA developers outline the intended outcomes using the MTA model. This model consists of MTA modules, MTA resources, and the interdependencies between them. Afterward, the SAP Cloud Deployment service validates it, orchestrates the necessary actions, and automates the MTA deployment. The outcome is the formation of Cloud Foundry applications, services, and the generation of SAP specific content. For more information about the Multitarget Application model, see the official specification documents [The Multitarget Application Model v.2](#) and [The Multitarget Application Model v.3](#).

You can create and deploy a Multitarget Application in the Cloud Foundry environment as described below by following different approaches that can yield the same result:

- Using SAP Web IDE Full-Stack as described in [Developing Multitarget Applications](#) - both the development descriptor `mta.yaml` and the deployment descriptor `mtad.yaml` are created automatically. The `mta.yaml` is generated when you create the application project, and the `mtad.yaml` file is created when you build the project.

#### ⓘ Note

You may still need to edit the development descriptor.

Development descriptors are used to generate MTA deployment descriptors, which define the required deployment data. That is, the MTA development descriptor data specifies what you want to build, how to build it, while the deployment descriptor data specifies as what and how to deploy it.



- <https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/a71bf8281254489ea8be6e323199b304.html> [https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/a71bf8281254489ea8be6e323199b304.html]
- <https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/3b533e3723674fad90f94510b92f10af.html> [https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/3b533e3723674fad90f94510b92f10af.html]
- <https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/1b0a7a0938944c7fac978d4b8e23a63f.html> [https://help.sap.com/viewer/825270ffffe74d9f988a0f0066ad59f0/CF/en-US/1b0a7a0938944c7fac978d4b8e23a63f.html]
- Using Business Application Studio - the flow is similar to WebIDE but the development is done from BAS.



- <https://help.sap.com/docs/bas/sap-business-application-studio/mta-development/> [https://help.sap.com/docs/bas/sap-business-application-studio/mta-development/]
- <https://help.sap.com/docs/bas/sap-business-application-studio/building-and-deploying-multitarget-applications> [https://help.sap.com/docs/bas/sap-business-application-studio/building-and-deploying-multitarget-applications]
- <https://help.sap.com/docs/bas/sap-business-application-studio/building-and-deploying-multitarget-applications> [https://help.sap.com/docs/bas/sap-business-application-studio/building-and-deploying-multitarget-applications]
- Using the [Cloud MTA Build Tool](#). Afterward, you deploy the MTA using the Cloud Foundry Command Line Interface.

#### ⓘ Note

An MTA development descriptor `mta.yaml` is required. You have to create it manually.



- <https://sap.github.io/cloud-mta-build-tool/> [https://sap.github.io/cloud-mta-build-tool/]
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/65ddb1b51a0642148c6b468a759a8a2e.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/65ddb1b51a0642148c6b468a759a8a2e.html]
- <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/c2d31e70a86440a19e47ead0cb349fdb.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/c2d31e70a86440a19e47ead0cb349fdb.html]
- Manually - create the required files manually and deploy them using the Cloud Foundry Command Line Interface

### ⓘ Note

An MTA development descriptor `mta.yaml` is not explicitly required. In this case, a deployment descriptor is maintained instead.

1. Create the required files using an IDE of your choice.
2. To produce your custom artifacts, use a build technology of your choice.
3. Maintain your deployment descriptor document (`mtad.yaml`).
  1. (Optional) If you want to have an MTA archive package, use the `mbt assemble`. For more information, see [Cloud MTA Build Tool \(MBT\)](#).
  2. Use the `cf deploy` command to deploy the MTA package or directly from the build result directory.

#### To learn more about

#### See

|   |  |
|---|--|
| Multitarget Application deployment descriptor | <a href="#">Defining Multitarget Application Deployment Descriptors for Cloud Foundry [page 578]</a>     |
| Multitarget Application archive               | <a href="#">Defining Multitarget Application Archives [page 576]</a>                                     |
| Multitarget Application extension descriptor  | <a href="#">Defining MTA Extension Descriptors [page 631]</a>  |
| Multitarget Application structure             | <a href="#">Multitarget Application Structure [page 584]</a>   |
| How to deploy the Multitarget Application     | <a href="#">Multitarget Application Plug-In for the Cloud Foundry Command Line Interface [page 2454]</a> |

## Terms and Concepts

### Terms and Concepts

| Term                          | Description  |
|-------------------------------|--|
| Multitarget application (MTA) | An application comprised of multiple software modules, which are created with different technologies and deployed to different runtimes. |

| Term                   | Description  |
|------------------------|--|
| Development descriptor | A YAML file named <code>mta.yaml</code> that contains a list of all entities, such as modules, resources, and properties that belong to an application or are used by it at runtime, and the dependencies between them. It is automatically generated when an MTA project is created or modified, or when a module is added or removed. The developer needs to edit the descriptor manually to define resources, properties, and dependencies, as well as fill in missing information. |
| Deployment descriptor  | A YAML file named <code>mtad.yaml</code> that contains a list of all entities which is created from either SAP Web IDE Full-Stack, SAP Business Application Studio, Cloud MTA Build Tool or manually. This file is similar to Development Descriptor but is used from the SAP Cloud Deployment service.  |
| Module                 | A self-contained application of a certain type, which is developed, packaged and deployed.   |
| Module type            | A type that defines the structure and the development technology of a module. You can see a list of the module types at <a href="#">Modules [page 595]</a> .   |
| Resource               | Any resource, such as an external service that is required by a module at runtime but not provided by the module itself.   |
| Property               | A property (key-value pair) of an application, module, or resource, that is used during deployment or at runtime.  |
| Parameter              | A reserved variable belonging to a module or resource, whose value is used during deployment or at runtime.  |
| Dependency             | <p>A relationship between a module and another module, resource, or property, such as provides and requires.</p> <ul style="list-style-type: none"> <li>• <code>provides</code>: indicates the properties or parameters that are provided by a module or resource to other modules.</li> <li>• <code>requires</code>: indicates other modules or resources that are required by a module in order to run.</li> </ul>   |
| MTA archive (MTAR)     | Archive containing a deployment descriptor, the module and resource binaries, and configuration files. The archive follows the JAR file specification.   |

## Prerequisites and Restrictions

You have to consider the following limits for the MTA artifacts, which can be handled by the SAP Cloud Deployment service:

- Maximum size of the MTA archive: 4 GB
- Maximum size of MTA module content: 1 GB
- Maximum size of MTA resource content: 1 MB

- Maximum size of MTA descriptors (`mtad.yaml` and `MANIFEST.MF`): 1 MB
- Maximum size of deployments which use multiple MTA archives: 20GB. This restriction is applicable only in transport scenarios via cTMS or CTS+, where one transport request with multiple MTAs results in a single deployment process in SAP Cloud Deployment service.

## Unsupported Cloud Foundry Features

In addition to the features that are not supported by the Cloud Foundry environment on SAP BTP (See [Supported and Unsupported Cloud Foundry Features](#)), MTA deployment in Cloud Foundry currently does not support these features as well:

- Sidecars. See <https://docs.cloudfoundry.org/devguide/sidecars.html>.
- Route services. See <https://docs.cloudfoundry.org/services/route-services.html>.
- Readiness health check. See <https://docs.cloudfoundry.org/devguide/deploy-apps/healthchecks.html>.
- Health check interval. See <https://docs.cloudfoundry.org/devguide/deploy-apps/healthchecks.html>.
- Metadata. See <https://docs.cloudfoundry.org/adminguide/metadata.html>.

### Note

These features might be implemented in the future. For the latest information, follow the release notes regarding Multitarget Applications for Cloud Foundry in [What's New for SAP Business Technology Platform](#).

## Related Information

[JAR File Specification](#)

[Cloud MTA Build Tool](#)

[Release Notes for Multitarget Applications for Cloud Foundry](#)

## 4.1.8.1.1 Benefits of the Multitarget Application Model for the Cloud Foundry Environment

The benefits of the multitarget application (MTA) approach can be related to the three “A”s that the MTA offers - Abstraction, Automation, and Assembly.

- Abstraction - Simply put, the MTA model uses abstraction to handle the diversity and complexity of different parts of an application, which may increase over time. This method offers a standardized way of modeling dependencies and environment-specific configurations. Without the MTA model you would have to individually handle all this heterogeneity, using custom scripts in a CI/CD tool to manage all underlying components, service dependencies and configurations. Achieving the needed automation through such custom scripts may be labor-intensive. It also demands higher effort as the technologies and services used by the application change over time. Modeling your application as an MTA highly simplifies the lifecycle of such heterogeneous applications.
- Automation - By defining your application's structure and dependencies (the “What”), you can use the SAP Cloud Deployment Service to manage automated tasks like deployment, service creation, updates and

binding (the “How”). This reduces the work required in CI/CD pipelines to oversee all these deployment-related tasks.

- Assembly - By offering a standard packaging format for your applications, the MTA serves as a release artifact. This is known as an assembly. The assembly, constituting the application and all its metadata, can be useful in different contexts. One such context is audited environments, where a signed artifact is mandatory. Another use case is moving deployable artifacts through a firewall. The assembly can also be valuable in situations where different policies are applicable based on the deployment target. Furthermore, this artifact can be distributed to other consumers. For instance, a partner distributing a business application to several customers may find this useful. These customers can then deploy this application in their own accounts or tenants.

## Deciding on the Right Size for a Multitarget Application

Your business application may grow over time. When this happens, you should decide on the volume of data for your MTA or multiple MTAs by analyzing which pieces make sense to be managed together as a separate lifecycle management unit.

A popular example is the case for a “core” or “framework” application parts that could work on their own, offering the basic business capabilities. Then there might be “extension” or “plugin” applications that extend the basic business capabilities, but are not mandatory and could be released and updated separately.

Another possibility is to model just one single Cloud Foundry application in a separate MTA. This makes sense if the application can be considered a self-contained microservice. However, even one Cloud Foundry app in one MTA modeling bears the benefits of dependency management (for example, backing service creation and external API lookup), content management (Fiori Launchpad configurations, workflow definitions, and so on) and configuration management (utilizing system placeholders, default and deployment specific configuration).

## Other Benefits

- Parallel deployment - This feature enables asynchronous deployment of multiple applications, this way the deployment of the archive is faster.
- Asynchronous service instance creation - The MTA deployer creates services asynchronously, this way decreasing the deployment time.
- Parallel undeployment - This feature enables asynchronous undeployment of applications, this way decreasing the undeployment time.

### 4.1.8.1.2 Getting Started

To learn how to deploy a simple multitarget application (MTA) to SAP Business Technology Platform, Cloud Foundry environment using the MultiApps CF CLI Plugin, you can follow the steps in the tutorial [Deploy Your First Multitarget Application](#).

If you want to see more MTA examples, see [CF MTA Examples](#) on GitHub.

### 4.1.8.1.3 Defining Multitarget Application Development Descriptors

Multitarget Applications are defined in a development descriptor required for design-time purposes.

The development descriptor (`mta.yaml`) defines the elements and dependencies of a Multitarget Application (MTA) compliant with the Cloud Foundry environment.

#### ⓘ Note

The MTA development descriptor (`mta.yaml`) is used to generate the deployment descriptor (`mtad.yaml`), which is required for deploying an MTA to the target runtime. If you use command-line tools to deploy an MTA, you do not need an `mta.yaml` file. However, in these cases you have to manually create the `mtad.yaml` file.

For more information about the MTA development descriptor, see [Defining Multitarget Application Deployment Descriptors for Cloud Foundry](#).

### Related Information

[Developing Multitarget Applications](#)

### 4.1.8.1.4 Defining Multitarget Application Archives

You package the MTA deployment descriptor and module binaries in an MTA archive. You can manually do so as described below, or alternatively use the Cloud MTA Build tool.

#### ⓘ Note

There could be more than one module of the same type in an MTA archive.

The Multitarget Application (MTA) archive is created in a way compatible with the JAR file specification. This allows us to use common tools for creating, modifying, and signing such types of archives.

#### ⚠ Caution

The maximum size of an MTA archive is limited to 500 MB. Deployment is denied for archives with larger size.

An MTA archive consists of the following:

- The `MANIFEST.MF` file
- The `mtad.yaml` MTA deployment descriptor file
- Application binaries, content, or configuration files

### ⓘ Note

- The MTA extension descriptor is not part of the MTA archive. During deployment you provide it as a separate file, or as parameters you enter manually when the SAP BTP requests them.
- Using a `resources` directory, as shown in some examples, is not mandatory. You can store the necessary resource files on root level of the MTA archive, or in another directory with name of your choice.

The following example shows the basic structure of an MTA archive. It contains a Java application `.war` file and a `META-INF` directory, which contains an MTA deployment descriptor with a module and a `MANIFEST.MF` file.

### ⚒ Example

```
/example.war  
/META-INF  
/META-INF/mtad.yaml  
/META-INF/MANIFEST.MF
```

The `MANIFEST.MF` file has to contain a name section for each MTA module part of the archive that has a file content. In the name section, the following information has to be added:

- Name - the path within the MTA archive, where the corresponding module is located. If it leads to a directory, add a forward slash (/) at the end.
- Content-Type - the type of the file that is used to deploy the corresponding module
- MTA-module - the name of the module as it has been defined in the deployment descriptor

### ⓘ Note

- You can store one application in two or more application binaries contained in the MTA archive.
- According to the JAR specification, there must be an empty line at the end of the file.

Sample content of the `META-INF/MANIFEST.MF` file:

### ⚒ Example

```
Manifest-Version: 1.0  
Created-By: example.com  
Name: example.war  
Content-Type: application/zip  
MTA-Module: example-java-app
```

The example above instructs the SAP BTP to:

- Look for the `example.war` file within the root of the MTA archive when working with module `example-java-app`
- Interpret the content of the `example.war` file as an `application/zip`

### ⓘ Note

The example above is incomplete. To deploy a solution, you have to create an MTA deployment descriptor. Then you have to create the MTA archive.

→ Tip

As an alternative to the procedure described above, you can also use the Cloud MTA Build Tool. See its official documentation at [Cloud MTA Build Tool](#).

## Related Information

[Cloud MTA Build Tool](#)

[The Multitarget Application Model v.2](#)

[The Multitarget Application Model v.3](#)

[JAR File Specification](#)

[Defining MTA Deployment Descriptors for the Neo Environment](#)

[Defining MTA Extension Descriptors \[page 631\]](#)

[MTA Module Types, Resource Types, and Parameters for Applications in the Neo Environment](#)

### 4.1.8.1.5 Defining Multitarget Application Deployment Descriptors for Cloud Foundry

The Multitarget Application (MTA) deployment descriptor is a YAML file that defines the relations between you as a provider of deployable artifacts and the SAP Cloud Deployment service in SAP BTP as a deployer tool.

Using the YAML data serialization language you describe the MTA in an MTA deployment descriptor (`mtad.yaml`) file following the [Multitarget Application Structure \[page 584\]](#).

ⓘ Note

As there are technical similarities between SAP HANA XS Advanced and Cloud Foundry, you can adapt application parameter for operation in either platform. Note that each environment supports its own set of module types, resource types, and configuration parameters. For more information, see the official specification documents [The Multitarget Application Model v.2](#) and [The Multitarget Application Model v.3](#).

## Schema Store

Writing `YAML` descriptors in plain text is often hard, so we have contributed a schema to the public Schema Store. This would provide you with an almost out-of-the-box support when writing MTA deployment descriptors in some of the more popular IDEs. The schema would provide you with auto-completion as well as suggestions and syntax checking.

For more information, see [Editor Schema Support](#).

## 4.1.8.1.5.1 MTA Deployment Descriptor Examples

Examples of MTA deployment descriptors.

### Note

The format and available options in the MTA deployment descriptor could change with the newer versions of the MTA specification. Always specify the schema version when defining an MTA deployment descriptor, so that the SAP BTP is aware against which specific MTA specification version you are deploying.

### Example 1

Deployment descriptor with a resource (managed service)

#### Example

```
_schema-version: "3.1.0"
ID: simple-mta
version: 1.0.0
modules:
- name: anatz
  type: javascript.nodejs
  requires:
    - name: hdi-service
resources:
- name: hdi-service
  type: org.cloudfoundry.managed-service
parameters:
  service: hana
  service-plan: hdi-shared
```

### Example 2

Deployment descriptor with the `keep-existing` parameter (will keep the existing: environment, service-bindings and routes)

#### Example

```
_schema-version: 3.1.0
ID: anatz-keep-existing
version: 4.0.0
modules:
- name: anatz
  type: staticfile
  path: hello-world.zip
parameters:
  memory: 64M
  route: new-custom-route-${space}
keep-existing:
  env: true
```

```
service-bindings: true
routes: true
```

## Example 3

Deployment descriptor with enabled parallel deployment of modules (will deploy modules in parallel)

### • Example

```
_schema-version: 3.1.0
ID: hello-world
version: 1.0.0
parameters:
  enable-parallel-deployments: true
modules:
- name: hello-world
  type: staticfile
  path: content/hello_world.zip
- name: hello-world-first
  type: staticfile
  path: content/hello_world.zip

- name: hello-world-second
  type: staticfile
  path: content/hello_world.zip
- name: hello-world-third
  type: staticfile
  path: content/hello_world.zip
```

## Example 4

Deployment descriptor with docker image module

### • Example

```
_schema-version: "3.1.0"
ID: docker-mtar
version: 2.0.1
modules:
- name: docker-image
  type: application
parameters:
  docker:
    image: cloudfoundry/test-app
```

## Example 5

Deployment descriptor with optional resource

### Example

```
_schema-version: "3.1.0"
ID: ztanal
version: 1.1.0
modules:
  - name: ztana
    type: javascript.nodejs
    requires:
      - name: test-resource

resources:
  - name: test-resource
    type: org.cloudfoundry(existing-service
    optional: true
    parameters:
      service: non-required-service
```

## Example 6

A basic MTA deployment descriptor that is defined in an `mtad.yaml` file:

### Example

```
_schema-version: "3.1"
ID: com.sap.xs2.samples.javahelloworld
version: 0.1.0

modules:
  - name: java-hello-world
    type: javascript.nodejs
    path: web/
    requires:
      - name: java-uaa
      - name: java
        group: destinations
        properties:
          name: java
          url: ~{url}
          forwardAuthToken: true
  - name: java-hello-world-backend
    type: java.tomee
    path: java/target/java-hello-world.war
    provides:
      - name: java
        properties:
          url: ${default-url}
    properties:
      JBP_CONFIG_RESOURCE_CONFIGURATION: "[ 'tomee/webapps/ROOT/WEB-INF/
resources.xml' : { 'service_name_for_DefaultDB' : 'java-hdi-container' } ]"
    requires:
      - name: java-uaa
      - name: java-hdi-container
```

```

      - name: java-hello-world-db
- name: java-hello-world-db
  type: com.sap.xs.hdi
  path: db/
  requires:
    - name: java-hdi-container

resources:
- name: java-hdi-container
  type: com.sap.xs.hdi-container

- name: java-uaa
  type: com.sap.xs.uaa-space
  parameters:
    config-path: xs-security.json

```

### ⓘ Note

The example above is incomplete. To deploy a solution, you have to create an MTA extension descriptor with the user and password added there. You also have to create the MTA archive.

## Example 7

Another basic example of an MTA deployment descriptor.

### ↗ Sample Code

```

_schema-version: "3.1"
ID: com.sap.xs2.samples.nodehelloworld
version: 0.1.0
modules:
- name: node-hello-world
  type: javascript.nodejs
  path: web/
  requires:
    - name: nodejs-uaa
    - name: nodejs
      group: destinations
      properties:
        name: backend
        url: ~{url}
        forwardAuthToken: true
      properties-metadata:
        name:
          optional: false
          overwritable: false
        url:
          overwritable: false
  parameters:
    host: !sensitive ${user}-node-hello-world
    memory: 128MB
  parameters-metadata:
    memory:
      optional: true
      overwritable: true
- name: node-hello-world-backend
  type: javascript.nodejs
  path: js/
  provides:
    - name: nodejs

```

```

properties:
  url: "${default-url}"
requires:
  - name: nodejs-uaa
  - name: nodejs-hdi-container
  - name: node-hello-world-db
parameters:
  host: ${user}-node-hello-world-backend
- name: node-hello-world-db
  type: com.sap.xs.hdi
  path: db/
  requires:
    - name: nodejs-hdi-container
parameters:
  tasks:
    - name: task-1           #You can model Cloud Foundry tasks
as described here. For additional information, check the Cloud Foundry
"Information for Developers" document.
  command: node deploy.js
  disk-quota: 1GB
  memory: 1GB
resources:
  - name: nodejs-hdi-container
    type: com.sap.xs.hdi-container
  parameters:
    config:
      schema: ${default-container-name}

  - name: nodejs-uaa
    type: com.sap.xs.uaa
  parameters:
    config-path: xs-security.json
  - name: log
    type: application-logs
    optional: true

```

## Example 8

A more complex example, which shows an MTA deployment description with the following modules:

- A database model
- An SAP UI5 application (hello world)
- An application written in node.js

The UI5 application “hello-world” uses the environment variable `<ui5_library>` as a logical reference to some version of UI5 on a public Website.

### Sample Code

```

ID: com.acme.xs2.samples.javahelloworld
version: 0.1.0
modules:
  - name: hello-world
    type: javascript.nodejs
    requires:
      - name: uaa
      - name: java_details
    properties:
      backend_url: ~{url3}/
properties:

```

```

ui5_library: "https://sapui5.hana.acme.com/"

- name: java-hello-world-backend
  type: java.tomee
  requires:
    - name: uaa
    - name: java-hello-world-db           # deploy ordering
    - name: java-hdi-container
  provides:
    - name: java_details
      properties:
        url3: ${url}                      # the value of the place-holder $
{url}
                                                # will be made known to the deployer

- name: java-hello-world-db
  type: com.sap.xs.hdi
  requires:
    - name: java-hdi-container
resources:
- name: java-hdi-container
  type: com.sap.xs.hdi-container

- name: java-uaa
  type: com.sap.xs.uaa
parameters:
  name: java-uaa                         # the name of the actual service

```

#### 4.1.8.1.6 Multitarget Application Structure

The following chapter contains information about:

- Global elements - an identifier and version that uniquely identify the MTA, including additional optional information such as a description, the providing organization, and a copyright notice for the author.
- [Modules \[page 595\]](#) - the deployable parts contained in the MTA deployment archive, most commonly Cloud Foundry applications or content.
- [Resources \[page 585\]](#) - they contain properties of resource types, which are entities not part of an MTA, but required by the modules at runtime or at deployment time.
- Dependencies between modules and resources.
- Parameters - variables which belong to a module or a resource, whose value is used during the deployment or at runtime. For more information, see [Parameters and Properties \[page 619\]](#).
- Properties - these result in the application environment variables that have to be available to the respective module at runtime. For more information, see [Parameters and Properties \[page 619\]](#).
- Technical configuration parameters, such as URLs, and application configuration parameters such as environment variables. For more information, see [Parameters and Properties \[page 619\]](#).
- Metadata - provide additional information about the declared parameters and properties. For more information, see [Metadata for Properties and Parameters \[page 630\]](#).
- [Module Hooks \[page 615\]](#) - use hooks to change the typical deployment process, for example to set them to be executed before or after the actual deployment steps for a module.
- If needed, use alternative arrangement of the properties' syntax. For more information see [Parameters and Properties \[page 619\]](#).

## 4.1.8.1.6.1 Resources

The application modules defined in the “modules” section of the deployment descriptor may depend on **resources**.

The resources may be used as:

- platform services managed by the deployer or only used by the applications
- configuration entries used by the applications and services

In the `resources` section, the following elements are mandatory:

- `name` - Must be unique within the MTA it identifies

Optional resource attributes include:

- `type` - the resource **type** is one of a reserved list of resource types supported by the MTA-aware deployment tools, for example: `com.sap.xs.uaa`, `com.sap.xs.hdi-container`, `com.sap.xs.job-scheduler`; the **type** indicates to the deployer how to discover, allocate, or provision the resource, for example, a managed service such as a database, or a user-provided service. When `type` is not defined, `resource` is used for configurations only for other modules and resources.
- `description` - non-translatable, free-text string; the string is not meant to be presented on application user interfaces (UI)
- `properties` - a structured set of name-value pairs
- `parameters` - reserved variables that affect the behavior of the MTA-aware tools, such as the deployer
- `active` - its value can be `true` or `false` and the default value is `true`. If set to `false`, the resource is not processed and it is ignored in the `requires` list of the application that requires it.
- `processed-after` - the attribute is used to create an order, in which resources should be processed. If a resource has this attribute, it will be processed after the other resources in a higher position are processed. The attribute value is a structured set of a list comprised of other resource names of the same MTA.

### → Tip

By default resources process in parallel, but if you want to enable an order of resources, see [Sequential Resource Processing \[page 674\]](#).

- `optional` - its value can be `true` or `false` and the default value is `false`. If set to `true`, the resource processing is fail-safe.

## Active and Inactive Resources

If you deploy an application with a resource type attribute `active` set to `false`, the resource is not provisioned and no binding is created. If you have already deployed the application with the resource type attribute `active` set to `true`, the binding to the resource is removed.

### Deployment with managed-service, existing-service and user-provided service

No binding is created and in case of `managed-service`, no service is created.

### Deployment with org.cloudfoundry(existing-service-key

No binding to an application environment will be done in the following cases:

- If the `org.cloudfoundry.existing-service-key` resource itself is set to `active: false`
- If the `org.cloudfoundry.existing-service-key` resource is set to `active: true`, but refers to a resource, which is set to `active: false`

### Deployment with configuration resources (cross MTA dependencies)

If the configuration resource is set to `active: false`, then:

- If the `requires` section of the module type expects a list, then the environment variable assigned to this list is empty and subscriptions between the modules is not created
- If the `requires` section does not expect a list, then no environment variable is created

See section “Optional resources” below for more information.

#### **Restriction**

System-specific parameters for the deployment descriptor must be included in a so-called MTA deployment extension descriptor.

## Optional Resources

To describe resources that are not mission-critical for the operation of your Multitarget Application, proceed as described below.

#### **Note**

This option is available with schema version 3.1.

You can declare some resources as optional, which mitigates the cases when they are not listed, not available, or have failed to be created or updated. This means that when the deployer cannot allocate the required resource due to any of these reasons, it generates a warning and continues processing. Alternatively, if a resource is not declared as optional, the deployer generates an error and stops processing.

The following excerpt is a code example for the `MANIFEST.MF` file.

#### **Sample Code**

```
...
resources:
...
- name: log
  type: com.sap.xs.auditlog
  optional: true
...
```

In the above example:

- If the `log` managed resource is not provided by the platform or landscape, a warning is logged and traced and the MTA deployment continues while ignoring the error.
- The available values for the `optional` parameter are `true` and `false`, with the latter being the default.

## Generic MTA Resource Types

- `org.cloudfoundry.managed-service`

In cases you have to choose a managed service and/or service plan that is not listed in [Generic MTA Resource Types \[page 587\]](#), you define it using the `org.cloudfoundry.managed-service` resource type with the following parameters:

- (Required) `service` - Name of the service to create.
- (Optional) `service-name` - Service instance name. Default value is the resource name.

### ⓘ Note

Service names that do not comply with the Cloud Foundry limitation of 50 symbols are automatically corrected. In such cases, the name is shortened and its end is replaced with a hash code.

- (Required) `service-plan` - Name of the service to create.

For example:

### ↔ Sample Code

```
resources:  
  - name: my-postgre-service  
    type: org.cloudfoundry.managed-service  
    parameters:  
      service: postgresql  
      service-plan: v9.6-dev
```

### ⓘ Note

To choose a different service plan for a predefined MTA resource type, for example to change the service plan for PostgreSQL service, you define it using:

### ↔ Sample Code

```
resources:  
  - name: my-postgre-service  
    type: org.postgresql  
    parameters:  
      service-plan: v9.6-dev
```

- `org.cloudfoundry.existing-service`

To indicate that the (named) service exists, without managing its lifecycle, you define the service name by using the `org.cloudfoundry.managed-service` resource type with the following parameters:

- (Optional) `service-name` - Service instance name. Default value is the resource name.

- `org.cloudfoundry.existing-service-key`

Existing service keys can be modeled as a resource of type `org.cloudfoundry.existing-service-key`, which checks and uses their credentials. For more information, see [Service Keys \[page 647\]](#).

- `org.cloudfoundry.user-provided-service`

Create or update a user-provided service configured with the following resource parameters:

- (Optional) `service-name` - Name of the service to create. Default value is the resource name.

### ⓘ Note

Service names that do not comply with the Cloud Foundry limitation of 50 symbols are automatically corrected. In such cases, the name is shortened and its end is replaced with a hash code.

- (Required) `config` - Map value, containing the service creation configuration, for example url and user credentials (user and password)

### ⚡ Example

#### ↔ Sample Code

```
resources:
  - name: my-destination-service
    type: org.cloudfoundry.user-provided-service
    parameters:
      config:
        <credential1>: <value1>
        <credential2>: <value2>
```

- `configuration`

For more information, see [Cross-MTA Dependencies \[page 652\]](#).

## Predefined MTA Resource Types

Modify the default MTA resource types by providing specific properties or parameters in the MTA deployment descriptor.

Predefined MTA Resource Types and Mapped Services

| Resource Type          | Service | Service Plan | Created Service |
|------------------------|---------|--------------|-----------------|
| com.sap.xs.hana-schema | hana    | schema       | Plain schema    |

### ⚡ Example

#### ↔ Sample Code

```
name: hcsp-sch
type:
com.sap.xs.hana-
schema
parameters:
service-name:
HCSP_SCH
config:
database_id:
9bf1f23a-123c-456e
-b789-ff12ea34a5dc
schema: HCSP_DB
```

| Resource Type               | Service | Service Plan | Created Service       |
|-----------------------------|---------|--------------|-----------------------|
| com.sap.xs.hana-securestore | hana    | securestore  | SAP HANA secure store |
| com.sap.xs.hdi-container    | hana    | hdi-shared   | HDI container         |

### ① Note

modules: When using the free trial subaccount, modify the default service:

### ↔ Sample Code

```
resources:
  - name: my-
    hdi-service
    type:
      com.sap.xs.hdi-
      container
    parameters:
      service:
        hanatrial
```

|                             |                  |             |   |
|-----------------------------|------------------|-------------|---|
| com.sap.xs.jobscheduler     | jobscheduler     | lite        | Job Scheduler   |
| com.sap.xs.uaa              | xsuaa            | application | Application UAA   |
| com.sap.xs.uaa-application  | xsuaa            | application | Application UAA   |
| com.sap.xs.uaa-devuser      | xsuaa            | application | Application UAA   |
| com.sap.xs.uaa-space        | xsuaa            | application | Application UAA   |
| com.sap.xs.application-logs | application-logs | lite        | Streams logs of bound applications to a central application logging stack |

### ⚠ Restriction

This resource type is now deprecated. Use `application-logs` instead.

|                             |                 |              |                 |
|-----------------------------|-----------------|--------------|-----------------|
| com.sap.portal.site-content | portal-services | site-content | Portal services |
|-----------------------------|-----------------|--------------|-----------------|

### ⚠ Restriction

Use only with the SAP Node.js module `@sap/site-content-deployer`

| Resource Type   | Service                | Service Plan   | Created Service  |
|---|------------------------|----------------|--|
| com.sap.portal.site-host  | portal-services        | site-host      | Portal services  |
| <b>⚠ Restriction</b>  |                        |                |  |
| Only for use with the SAP Node.js module <code>@sap/site-entry</code>   |                        |                |  |
| com.sap.xs.auditlog   | auditlog               | standard       | Audit log service  |
| auditlog  | auditlog               | standard       | Audit log service  |
| autoscaler  | autoscaler             | lite           | Automatically increase or decrease the number of application instances based on a policy you define. |
| application-logs  | application-logs       | lite           | Streams logs of bound applications to a central application logging stack                            |
| connectivity  | connectivity           | lite           | Establishes a secure and reliable connectivity between cloud applications and on-premise systems     |
| destination   | destination            | lite           | Provides a secure and a reliable access to destination configurations                                |
| feature-flags   | feature-flags          | lite           | Feature Flags service for controlling feature rollout  |
| ml-foundation-services  | ml-foundation-services | lite           |  |
| objectstore   | objectstore            | s3-standard    | Highly available and distributed consistent object store   |
| org.mongodb   | mongodb                | v3.0-container | MongoDB document-oriented database system.   |
| <b>⚠ Restriction</b>  |                        |                |  |
| This resource type is now deprecated. Use <code>mongodb</code> instead. |                        |                |  |
| mongodb   | mongodb                | v3.0-container | MongoDB document-oriented database system  |

| Resource Type        | Service  | Service Plan   | Created Service                              |
|----------------------|--|----------------|--|
| org.postgresql       | postgresql   | v9.4-container | PostgreSQL object-relational database system |
| <b>⚠ Restriction</b> | This resource type is now deprecated. Use <code>postgresql</code> instead. |                |  |
| postgresql           | postgresql   | v9.4-container | PostgreSQL object-relational database system |
| com.rabbitmq         | rabbitmq   | v3.6-container | RabbitMQ messaging                           |
| <b>⚠ Restriction</b> | This resource type is now deprecated. Use <code>rabbitmq</code> instead.   |                |  |
| rabbitmq             | rabbitmq   | v3.6-container | RabbitMQ messaging                           |
| io.redis             | redis  | v3.0-container | Redis in-memory data structure store         |
| <b>⚠ Restriction</b> | This resource type is now deprecated. Use <code>redis</code> instead.      |                |  |
| redis                | redis  | v3.0-container | Redis in-memory data structure store         |

## Resource-Specific Parameters

Resource parameters have platform-specific semantics. To reference a parameter value, use the placeholder notation `$(<parameter>)`, for example, `$(default-host)`.

### → Tip

It is also possible to declare metadata for parameters and properties defined in the MTA deployment description; the mapping is based on the parameter or property keys. For example, you can specify if a parameter is **required** (optional: false) or can be modified overwritable: true.

The following parameters are supported:

### ⓘ Note

If you can't find a specific parameter from the native Cloud Foundry manifest here, refer to [Prerequisites and Restrictions \[page 573\]](#) to see which Cloud Foundry features are currently not supported.

## MTA Development and Deployment Parameters

| Parameter              | Read-Only<br>(System) | Description  | Default Value   | Example   |
|------------------------|-----------------------|--|---|---|
| apply-namespace        | Write                 | Applies namespace to the service name. If the namespace value is not provided in the CLI options, it is not applied. For more information, see <a href="#">Fine-Grained Configuration [page 667]</a> .   |   | <pre>resources: - name: java   ...   parameters:     apply-namespace: true</pre>  |
| config                 | Write                 | Defines service creation parameters. More information in <a href="#">Service Creation Parameters [page 641]</a> .  | n/a   | See <a href="#">Service Creation Parameters [page 641]</a> .  |
| syslog-drain-url       | Write                 | The URL to which logs for bound applications are streamed.   | n/a   | <pre>resources: - name: service-name   type: org.cloudfoundry.user-provided-service   parameters:     syslog-drain-url: syslog://example.log-aggregator.com</pre> |
| default-container-name | Yes                   | Default value for the container-name parameter that is used during HDI creation. It is based on the organization, space and resource name, which are combined in a way that conforms to the container-name restrictions for length and legal characters. All dash (-) symbols are replaced with an underscore (_). | Generated as described in the description.            | <org_name>_<space_name>_<resource_name>   |
| default-service-name   | Yes                   | The name of the service in the Cloud Foundry environment to be created for this resource, based on the resource name with or without a name-space prefix.  | The resource name with or without a name-space prefix | nodejs-hdi-container com.sap.xs2.samples.xsj shellworld.nodejs-hdi-container  |

| Parameter  | Read-Only<br>(System) | Description   | Default Value                              | Example  |
|--|-----------------------|---|--|--|
| default-xsappname  | Yes                   | Default value for the <code>xsappname</code> parameter that is used during UAA creation. It is based on the service name, which is modified in a way that conforms to the <code>xsappname</code> restrictions for length and legal characters.  | Generated as described in the description. | <code>xs-deploy-service-database</code> (if the service name is “ <code>xs@-deploy-service-database</code> ”)  |
| service  |                       | The type of the created service   | Empty, or as specified in resource-type    | <code>service: hana</code>   |
| service-key-name   | Write                 | Used when consuming an existing service key. Specifies the name of the service key. See <a href="#">Consumption of Service Keys [page 647]</a> for more information.  | The name of the resource.                  | <code>service-key-name: my-service-key</code>  |
| service-name   |                       | The name of the service in the Cloud Foundry environment to be created for this resource, based on the resource name with or without a name-space prefix.   | <code>\${default-service-name}</code>      | <code>nodejs-hdi-container</code><br><code>com.sap.xs2.samples.xsj</code><br><code>shellworld.nodejs-hdi-container</code>                                    |
| <b>① Note</b>  |                       |   |  |  |
| Service names that do not comply with the Cloud Foundry limitation of 50 symbols are automatically corrected. In such cases, the name is shortened and its end is replaced with a hash code. |                       |   |  |  |
| service-plan   |                       | The plan of the created service   | Empty, or as specified in resource-type    | <code>service-plan: hdi-shared</code>  |
| service-tags   | Write                 | Some services employ a list of custom tags, which provide an easier way for applications to parse <code>&lt;VCAP_SERVICES&gt;</code> for credentials. You can provide custom tags when creating a service instance. For more information, see <a href="#">Service Tags [page 645]</a> . | n/a  | <pre>resources:   - name: nodejs-uaa     type: com.sap.xs.uaa     parameters:       service-tags:         [ "custom-tag-A",           "custom-tag-B" ]</pre> |

| Parameter            | Read-Only<br>(System) | Description  | Default Value   | Example  |
|----------------------|-----------------------|--|---|--|
| service-broker       |                       | Use this parameter to specify the service broker you want to employ when you create your service. This can be useful for testing purposes, among others.   | Name of service broker you want to be used.   | <pre>resources:   - name: test-service     type: org.cloudfoundry.managed-service     parameters:       service: foo       service-plan: bar-a       service-broker: test-broker-1</pre>   |
| skip-service-updates |                       | This parameter allows you to specify the service configuration changes to ignore, when deciding if you want to update a service instance – in particular changes of service parameters, plan, or tags. | The default value for all is false. The service instance would be updated on change in any of the configurations. | <pre>resources:   - name: service-name     type: org.cloudfoundry.managed-service     parameters:       skip-service-updates:         plan: true       parameters: true       tags: true</pre> <p>In the example above, skip-service-updates modifies the update strategy as follows:</p> <ul style="list-style-type: none"> <li>• allows users to specify which configurations should not be updated</li> </ul> <p>Note that these 3 key-value pairs can be in any order.</p> |

#### → Tip

For a better understanding of the interactions among the service, service broker, and service instances, see [Provisioning and integrating service instances](#).

## Related Information

[Services Overview](#)

[Provisioning and integrating service instances](#)

## 4.1.8.1.6.2 Modules

The `modules` section of the deployment descriptor lists the deployable parts contained in the MTA deployment archive.

The following elements are mandatory:

- `name` - must be unique within the MTA it identifies
- `type` - defines which deployment mechanism to be used for this module

Optional module attributes include:

- `path` - the file-system path relative to the root of the MTA directory. The content of the file is used to create or update the CF app or content, depending on the module target. The `path` is used only during MTA build. In an already built MTA archive (MTAR), path is ignored and only corresponding entry in `MANIFEST.MF` for the module is processed.  
The path is applicable only when MTA is assembled based on deployment descriptor (`mtad.yaml`), and not on development descriptor (`mta.yaml`)
- `description` - non-translatable, free-text string; the string is not meant to be presented on application user interfaces (UI)
- `properties` - a structured set of name-value pairs; if a module, which requires the resource, represents a CF application, the resource properties are injected into the environment of the application
- `parameters` - reserved variables that affect the behavior of the MTA-aware tools, such as the deployer
- `deployed-after` - the attribute is used to create an order, in which modules should be processed. If a module has this attribute, it will be processed after the other modules in a higher position are processed. The attribute value is a structured set of a list comprised of other module names of the same MTA.
- `requires` - specifies the names of `requires` sections provided in `resource` that have been declared for the same MTA. Tools check if all required names are provided within the MTA.
- `provides` - specifies the names of `provides` sections, each containing configuration data; the data provided can be `required` by other modules in the same MTA

→ Tip

Modules can be deployed in parallel. See [Parallel Module Deployment \[page 672\]](#).

## MTA Module Types

Modify the following MTA module types by providing specific properties or parameters in the MTA deployment descriptor (`mtad.yaml`).

## MTA Default Modules Types

| Module Type        | Default Parameter Values and Description   | Module Properties | Result  |
|--------------------|--|-------------------|---|
| javascript.no.dejs | No default parameter.  | None              | CF application with automatic buildpack detection                     |
|                    | <p><b>① Note</b></p> <p>We strongly recommend that you specify the appropriate buildpack for your application.modules: parameters: buildpack: binary_buildpack To do so, use the buildpack - name: my-binary-app module parameter.</p> |                   |   |
| nodejs             | buildpack(nodejs_buildpack)  | None              | CF application with Node.js runtime                                   |
| custom             | No default parameter.  | None              | CF application with automatic buildpack detection                     |
|                    | <p><b>① Note</b></p> <p>We strongly recommend that you specify the appropriate buildpack for your application. To do so, use the buildpack module parameter.</p>   |                   |   |
| application        | No default parameter.  | None              | CF application with automatic buildpack detection                     |
|                    | <p><b>① Note</b></p> <p>We strongly recommend that you specify the appropriate buildpack for your application. To do so, use the buildpack module parameter.</p>   |                   |   |
| java               | buildpack(sap_java_buildpack)  | None              | CF application with Automatic runtime detection by sap_java_buildpack |
| java.tomcat        | buildpack(sap_java_buildpack)  | None              | CF application with Tomcat runtime of sap_java_buildpack              |
| java.tomee7        | buildpack(sap_java_buildpack)  | tomee7            | CF application with TomEE 7 runtime of sap_java_buildpack             |

| Module Type                 | Default Parameter Values and Description  | Module Properties | Result  |
|-----------------------------|---|-------------------|---|
| com.sap.xs.hdi              | <ul style="list-style-type: none"> <li>• no-route (true)<br/>Do not assign a route to the application.</li> <li>• memory (256MB)</li> <li>• health-check-type - (none)</li> <li>• no-start (true)</li> <li>• cf-task</li> <li>• command (npm start)</li> <li>• memory (256MB)</li> <li>• name (deploy)</li> <li>• dependency-type (hard)<br/>In circular module-dependencies, deploy modules with dependency type "hard" first</li> <li>• buildpack(nodejs_buildpack)</li> </ul>  | EXIT (1)          | CF application with HDI content activation      |
| com.sap.xs.hdi-dynamic      | buildpack(nodejs_buildpack)   | None              | CF application with Node.js runtime             |
| com.sap.portal.content      | <p><b>① Note</b></p> <p>Before using this module type, update the content deployer applications to their latest version.</p> <ul style="list-style-type: none"> <li>• no-route (true). Defines if a route should be assigned to the application.</li> <li>• no-start - (true). Only the one-off tasks will be executed, that is, without triggering the start of the application.</li> <li>• memory (256M). Defines the memory allocated to the application.</li> <li>• tasks (name:deploy, memory:256M, command:npm start)</li> </ul> <p>For more information, see <a href="#">Tasks [page 639]</a>.</p> <ul style="list-style-type: none"> <li>• dependency-type(hard). Defines if this module should be deployed first, if it takes part in circular module dependency cycles. If hard means that this module is deployed first.</li> <li>• buildpack(nodejs_buildpack)</li> </ul> | None              | CF application with SAP Fiori launchpad content |
| com.sap.portal.site-content | <p>This module type is <b>deprecated</b>. You have to use com.sap.portal.content instead.</p>   | None              | CF application with SAP Fiori launchpad content |

| Module Type                 | Default Parameter Values and Description  | Module Properties | Result  |
|-----------------------------|---|-------------------|---|
| com.sap.application.content | <p>Required dependency parameters:</p> <ul style="list-style-type: none"> <li>• <code>content-target(false)</code></li> </ul> <p>Specify that the resource would be used as a target for the module content deployment.</p> | None              | Direct content deployment to backing services |

| Module Type                       | Default Parameter Values and Description   | Module Properties | Result   |
|-----------------------------------|--|-------------------|--|
| com.sap.html5.application-content | <p><b>① Note</b></p> <p>This module type is <b>deprecated</b>. Please use <code>com.sap.html5.application.content</code> instead.</p> <p><b>① Note</b></p> <p>Keep the dependency to <code>@sap/html5-app-deployer</code> updated to the latest version. If you are using an older version, you might encounter some issues.</p> <ul style="list-style-type: none"> <li>• <code>no-route(true)</code>. Defines if a route should be assigned to the application.</li> <li>• <code>memory(256M)</code>. Defines the memory allocated to the application.</li> <li>• <code>execute-app(true)</code>. Defines whether the application is executed. If yes, the application performs certain amount of work and upon completion sets a <code>success-marker</code> or <code>failure-marker</code> by means of a log message.</li> <li>• <code>success-marker(STDOUT:The deployment of html5 application content done.*)</code></li> <li>• <code>failure-marker(STDERR:The deployment of html5 application content failed.*)</code></li> <li>• <code>stop-app(true)</code>. Defines if the application should be stopped after execution.</li> <li>• <code>check-deploy-id(true)</code> - Defines if the deployment (process) ID should also be checked when checking the application execution status.</li> <li>• <code>dependency-type(hard)</code>. Defines if this module should be deployed first, if it takes part in circular module dependency cycles. If <code>hard</code> means that this module is deployed first.</li> <li>• <code>health-check-type(none)</code>. Defines if the module should be monitored for availability.</li> </ul> | None              | Deploys a content deployment application, and creates a task that performs the content deployment. |

| Module Type                       | Default Parameter Values and Description  | Module Properties | Result                                  |
|-----------------------------------|---|-------------------|---|
|                                   | <ul style="list-style-type: none"> <li>• buildpack(nodejs_buildpack)</li> </ul>   |                   |   |
| com.sap.html5.application.content | <p><b>① Note</b></p> <p>Keep the dependency to @sap/html5-app-deployer updated to the latest version. If you are using an older version, you might encounter some issues.</p>   | EXIT (1)          | CF application with SAP HTML5 content   |
|                                   | <ul style="list-style-type: none"> <li>• no-route(true). Defines if a route should be assigned to the application.</li> <li>• no-start - (true). Only the one-off tasks will be executed, that is, without triggering the start of the application.</li> <li>• memory(256M). Defines the memory allocated to the application.</li> <li>• tasks(name:deploy, memory:256M, command:npm start)</li> </ul> <p>For more information, see <a href="#">Tasks [page 639]</a>.</p> <ul style="list-style-type: none"> <li>• dependency-type(hard). Defines if this module should be deployed first, if it takes part in circular module dependency cycles. If hard means that this module is deployed first.</li> <li>• buildpack(nodejs_buildpack)</li> </ul> |                   |   |
| staticfile                        | buildpack(staticfile_buildpack)   | None              | CF application with static file runtime |
| go                                | buildpack(go_buildpack)   | None              | CF application with Go runtime          |
| python                            | buildpack(python_buildpack)   | None              | CF application with Python runtime      |
| php                               | buildpack/php_buildpack)  | None              | CF application with PHP runtime         |
| binary                            | buildpack(binary_buildpack)   | None              | CF application with Binary runtime      |

To choose a binary\_buildpack, define it by using the following:

#### ↔ Sample Code

```
modules:
  - name: my-binary-app
    type: custom
    parameters:
      buildpack: binary_buildpack
```

## Module-Specific Parameters

Module parameters have platform-specific semantics. To reference a parameter value, use the placeholder notation \${<parameter>}}, for example, \${default-host}.

### → Tip

It is also possible to declare metadata for parameters and properties defined in the MTA deployment description; the mapping is based on the parameter or property keys. For example, you can specify if a parameter is **required** (optional; false) or can be modified overwritable: true.

The following parameters are supported:

### ⓘ Note

If you can't find a specific parameter from the native Cloud Foundry manifest here, refer to [Prerequisites and Restrictions \[page 573\]](#) to see which Cloud Foundry features are currently not supported.

#### MTA Development and Deployment Module Parameters

| Parameter       | Read-Only (System) | Description  | Default Value  | Example  |
|-----------------|--------------------|--|--|--|
| app-name        | Read-Only (System) | The name of the application in the Cloud Foundry environment to be deployed for this module, based on the module name  | \${default-app-name}                                       | node-hello-world<br>com.sap.xs2.samples.xsj<br>shellworld.node-hello-world                       |
| apply-namespace | Write              | Applies namespace to the application name. When you set apply-namespace to the application name and do not specify apply-namespace for its route, the namespace is applied to both the application name and its route. If the namespace value is not provided in the CLI options, it is not applied. For more information, see <a href="#">Fine-Grained Configuration [page 667]</a> . |  | <pre>modules:<br/>- name: java<br/>...<br/>parameters:<br/>  apply-namespace:<br/>    true</pre> |
| buildpack       |                    | The name or the URL of a custom buildpack required by the application  | Empty, or as specified in the deploy service configuration | buildpack: git://github.acme.com/xs2-java/xs2javabuildpack                                       |
| buildpacks      |                    | An array of buildpacks. If a buildpack parameter already exists, it will be overwritten by the buildpacks listed in the buildpacks parameter, so that you have to include it in the array.   | Empty, or as specified in the deploy service configuration | buildpacks:<br>[java_buildpack,<br>nodejs_buildpack,<br>staticfile_buildpack]                    |

| Parameter             | Read-Only (System) | Description   | Default Value  | Example  |
|-----------------------|--------------------|---|--|--|
| command               |                    | A custom command required to start the application  | Empty, or as specified in the deploy service configuration | command: node index.js   |
| create-service-broker |                    | Specifies whether [true false] a service broker should be registered for the application module                       | false  | create-service-broker: true  |
| default-app-name      | Yes                | The name of the application in the Cloud Foundry environment to be deployed for this module, based on the module name | The module name with or without a name-space prefix        | node-hello-world<br>com.sap.xs2.samples.xsjs<br>shelloworld.node-hello-world |

| Parameter         | Read-Only (System) | Description  | Default Value                             | Example                        |
|-------------------|--------------------|--|---|--------------------------------|
| default-host      | Yes                | <p>The default host name, which is composed based on the module name to ensure uniqueness. Used with host-based routing to compose the default URI, see below. It follows the convention \${org}-\${space}-&lt;module_name&gt;.</p> <p><b>→ Tip</b><br/>We recommend you explicitly use the routes parameter. See <a href="#">Routes [page 649]</a>.</p>   | Generated as described in the description | trial-a007007-node-hello-world |
| default-instances | Yes                | <p>The number of application instances that are started during the deployment</p> <p><b>ⓘ Note</b><br/>Host names that do not comply with the Cloud Foundry naming limitations are automatically corrected:</p> <ul style="list-style-type: none"> <li>Any characters that are not within the allowed a–z and 0–9 ranges are replaced by dashes (-) to prevent deployment issues.</li> <li>If the host name length exceeds 63 symbols, the name is shortened and its end is replaced with a hash code.</li> </ul> <p><b>ⓘ Note</b><br/>When used in blue-green deployment the value is resolved depending on the phase. During the testing phase the "idle" suffix will be appended to the host. If you want to use the live version only, refer to \${default-live-host}.</p> | 1   | default-instances: 1           |

| Parameter             | Read-Only (System) | Description   | Default Value                                       | Example   |
|-----------------------|--------------------|---|---|---|
| default-live-app-name |                    | Valid for blue-green deployment. Specify this parameter if you want the name of the Cloud Foundry application that is to be deployed with this module to be based on the name of the module. For standard deployment the parameter will be the same as default-app-name regardless the phase. | The module name with or without a name-space prefix | node-hello-world  |
| default-live-domain   |                    | Valid for blue-green deployment. The value of this parameter is the default domain for the current organization.  |   |   |
| default-live-host     |                    | Valid for blue-green deployment. Specify this parameter if you want to use \${default-host} without the "idle" suffix during the testing phase of the blue-green deployment.  | Generated as described in the description           | trial-a007007-node-hello-world                          |
| default-live-uri      |                    | Valid for blue-green deployment. Specify this parameter if you want to use \${default-uri} without the "idle" suffix during the testing phase of the blue-green deployment.   | Generated as described in the description           | trial-a007007-node-hello-world.cfapps.acme.ondemand.com |
| default-live-url      |                    | Valid for blue-green deployment. Specify this parameter if you want to use \${default-url} without the "idle" suffix during the testing phase of the blue-green deployment.   | Generated as described in the description           | \${protocol}://\${default-live-uri}                     |
| default-uri           | Yes                | The default URI, composed as \${host}.\${domain} (host-based routing). Note that \${host} will be the same as \${default-host}, unless specified explicitly as a parameter. Similarly, \${domain} would be the same as \${default-domain}, unless specified explicitly.                       | Generated as described in the description.          | trial-a007007-node-hello-world.cfapps.acme.ondemand.com |

### ⓘ Note

When used in blue-green deployment the value is resolved depending on the phase. During the testing phase the "idle" suffix will be appended to the host of the URL. If you want to use the live version only, refer to \${default-live-uri}.

| Parameter       | Read-Only (System) | Description   | Default Value                              | Example   |
|-----------------|--------------------|---|--|---|
| default-uri     | Yes                | The default URL, composed as \${protocol}://\${default-uri}. Note that the \${default-uri} placeholder is resolved as \${host}.\${domain} (host-based routing)  | Generated as described in the description. | \${protocol}://\${default-uri}  |
|                 |                    | <p><b> ⓘ Note</b></p> <p>When used in blue-green deployment the value is resolved depending on the phase. During the testing phase the "idle" suffix will be appended to the host of the URL. If you want to use the live version only, refer to \${default-live-url}.</p>  |  |   |
| dependency-type |                    | Deployment order of modules with circular dependencies  | soft                                       | dependency-type: hard<br>dependency-type: soft  |
| disk-quota      |                    | The disk space that will be available to the application. This parameter requires a unit of measurement M, MB, G, or GB in upper or lower case.   | 1, or as specified in module-type          | disk-quota: 1G  |
| docker          | Write              | Creates a module from a docker image. When using a docker image parameter, we do not need to specify the module in the MANIFEST.mf file. An image parameter is a docker image from the Docker Hub or somewhere else. The username and the password are optional, but if a Docker image from a private repository is uploaded, then they are mandatory.<br><br>When uploading a docker image, the content of a module is not needed. | n/a  | <p><b> ⌂ Sample Code</b></p> <pre>modules:   name: foo   type:   application     parameters:       docker:         image:         cloudfoundry/test-app         username:         &lt;optional         username&gt;         password:         &lt;optional         password&gt;</pre> |
| domain          |                    | The domain on which the application is available later  | \${default-domain}                         | domain: \${default-domain}.acme.com   |

| Parameter                        | Read-Only (System) | Description  | Default Value  | Example  |
|----------------------------------|--------------------|--|--|--|
| domains                          |                    | The domains on which the application is available later. The resulting application routes are the Cartesian product of the domains and hosts. That is, a separate route for each host is constructed on each domain.           | domains: - \${default-domain}.acme.com<br>- test-\${default-domain}.acme.com                     | domains: - \${default-domain}.acme.com<br>- test-\${default-domain}.acme.com   |
| default-domain                   | Yes                | The value of this parameter is the default domain for the current organization.  |  |  |
| enable-ssh                       |                    | Enables use of SSH within an application. Supported for the Diego container runtime environment only.  | n/a  | "enable-ssh": true<br>"enable-ssh": false  |
| enable-parallel-service-bindings |                    | Enables or disables the parallel binding or unbinding of services during deployment. If disabled, the services are bound and unbound sequentially in the order provided in the deployment descriptor.                          | true   | <p><b>Sample Code</b></p> <pre>- name: &lt;module name&gt;   type: &lt;module type&gt;   parameters:     enable-parallel-service-bindings: false</pre> |
| health-check-http-endpoint       |                    | If the health-check-type parameter is set to http, the controller will do a GET request to this endpoint. The application will be considered as healthy if the response is 200 OK.   | If health-check-type is set to http, the default value is /, otherwise there is no default value | health-check-type: http<br>health-check-http-endpoint: /health   |
| health-check-timeout             |                    | The application health check timeout in seconds  | n/a  | health-check-timeout:120   |
| health-check-invocation-timeout  |                    | The time period in seconds, within which the application health check should be automatically started. If the health check does not start within the defined time period, it is omitted, and the deployment process continues. | n/a  | health-check-invocation-timeout:16   |
| health-check-type                |                    | The application health check type  | port   | health-check-type: port<br>health-check-type: http<br>health-check-type: process   |

| Parameter      | Read-Only (System) | Description  | Default Value                     | Example  |
|----------------|--------------------|--|-----------------------------------|--|
| host           |                    | <p>The hostname or subdomain where an application is available later.</p> <p>If you want to create a wildcard host name, use an asterisk in quotes ("*").</p>  | <code>#{default-host}</code>      | <pre>host: \${space}-node-hello-world host: "*"</pre>  |
| hosts          |                    | <p>The hostnames or subdomain where an application is available later.</p> <p>If you want to create a wildcard host name, use an asterisk in quotes ("*").</p>   | <code>hosts: - \${host}</code>    | <pre>modules: - name: my-app   type: application   parameters:     hosts:       - \${space}-node-hello-world       - "*"</pre>                       |
| idle-domain    |                    | <p>Valid for a blue-green deployment when a new application version is started on a temporary route.</p> <p>Specify this parameter if you want to use another domain for temporary routes.</p>                                   | <code>#{default-domain}</code>    | <pre>modules: - name: app   type: nodejs   parameters:     idle-domain: "&lt;some.domain&gt;"</pre>  |
| idle-domains   |                    | <p>Valid for a blue-green deployment when a new application version is started on several temporary routes.</p> <p>Specify this parameter if you want to use other domains for temporary routes by listing them in an array.</p> | <code>#{default-domain}</code>    | <pre>modules: - name: app   type: nodejs   parameters:     idle-domains:       [ "&lt;some.domain&gt;" ,         "&lt;some.other.domain&gt;" ]</pre> |
| idle-host      |                    | <p>Valid for a blue-green deployment when a new application version is started on a temporary route.</p> <p>Specify this parameter if you want to use another host for a temporary route.</p>                                    | <code>#{default-host}-idle</code> | <pre>modules: - name: app   type: nodejs   parameters:     idle-host: "&lt;some-hostname&gt;"</pre>  |
| <b> ⓘ Note</b> |                    | <p>The new application will start on a route comprised of the specified host and the default domain.</p>   |                                   |  |

| Parameter     | Read-Only (System) | Description  | Default Value                       | Example   |
|---------------|--------------------|--|-------------------------------------|---|
| idle-hosts    |                    | <p>Valid for a blue-green deployment when a new application version is started on temporary routes.</p> <p>Specify this parameter if you want to use other hosts for temporary routes by listing them in an array.</p>   | <code> \${default-host}-idle</code> | <pre>modules: - name: app   type: nodejs   parameters:     idle-     hosts: [ "&lt;some- hostname&gt;", "&lt;some- other-hostname&gt;" ]</pre>  |
|               |                    | <p><b> ⓘ Note</b></p> <p>The new application will start on routes comprised of the specified hosts and default domain.</p>   |                                     |   |
| idle-routes   |                    | <p>Valid for a blue-green deployment when a new application version is started on temporary routes.</p> <p>Specify this parameter if you want to use other routes for the application.</p>   | <code> \${default-uri}-idle</code>  | <pre>modules: - name: app   parameters:     idle-routes:       - idle-route:         "&lt;your-first-idle- hostname.your.first.i idle.domain&gt;"       - idle-route:         "&lt;your-second-idle- hostname.your.second. idle.domain&gt;"</pre> |
| instances     |                    | The number of application instances that will be started during the deployment   | <code> \${default-instances}</code> | <code> instances: 2</code>  |
| keep-existing |                    | Defines the application attributes which will be kept after the deployment or blue-green deployment has finished. The supported attributes which could be kept are application environment, application bindings and application routes. If not specified, the default values are false, which indicates that each application attribute will be updated with the new values presented in the deployment descriptor. | <code> false</code>                 | <p><b> ↴ Sample Code</b></p> <pre>keep-existing:   env: true   service-   bindings: true   routes: false</pre>  |

| Parameter            | Read-Only (System) | Description  | Default Value                        | Example  |
|----------------------|--------------------|--|--------------------------------------|--|
| keep-existing-routes | Write              | <p>When specified on module level, it indicates if the existing routes of the module's corresponding application should be kept even if they are not defined within the deployment and/or extension descriptors.</p> <p>When specified on global level, under the <code>parameters</code> section of the descriptor, it indicates if the existing routes of all applications within that MTA should be kept.</p>   | false                                | <p> <a href="#">Sample Code</a></p> <pre>parameters:   keep-existing-routes: true modules:   - name: foo     type: nodejs   parameters:     keep-existing-routes:       false       - name: bar         type: nodejs       - name: baz         type: nodejs</pre> |
| memory               |                    | <p> <b>Note</b></p> <ul style="list-style-type: none"> <li>The module-level variant of the parameter has priority over the global parameter.</li> <li>This parameter is typically used when users want to keep the routes they have mapped manually by using the <code>cfe map-route</code> command. We discourage this approach, as manual operations could lead to inconsistent deployment results and difficult troubleshooting. We recommend you to define all routes in the deployment and/or extension descriptors, which allows for their automatic management.</li> </ul> | 256M, or as specified in module-type | memory: 128M   |
| no-route             | Write              | Configures the deployer to create or skip the creation of a route for the application described by the module  | false                                | no-route: true   |

| Parameter             | Read-Only (System) | Description  | Default Value   | Example   |
|-----------------------|--------------------|--|---|---|
| no-start              |                    | <p>Start/do not start the application during deployment.</p> <p><b>→ Tip</b></p> <p>This parameter setting overrides the command-line option <code>--no-start</code>.</p> <p>If you explicitly set the <code>no-start</code> to <code>false</code> for the module <code>foo</code> in the example provided, then the module <code>foo</code> <b>is</b> started on deployment, even if you also specify the command-line option <code>--no-start</code> with the <code>cf deploy</code> command.</p>  | Depends on the command line option <code>--no-start</code>  | <code>no-start: true</code>   |
| restart-on-env-change | Write              | <p>Specifies whether an application should be restarted if an environment variable has been changed in one of the following categories:</p> <ul style="list-style-type: none"> <li>• <code>vcap-application</code></li> <li>• <code>vcap-services</code></li> <li>• <code>user-provided</code></li> </ul> <p><b>ⓘ Note</b></p> <p>If you set these parameters to <code>false</code>, the changes in environment are not consumable by a running instances of the application. If your application depends on the latest environment, it might become outdated.</p> | <pre>restart-on-env-change:   vcap-application: false   vcap-services: true   user-provided: true</pre> | <p><b>↔ Sample Code</b></p> <pre>restart-on-env-change:   vcap-application: false   vcap-services: true   user-provided: true</pre> |

| Parameter                   | Read-Only (System) | Description   | Default Value   | Example  |
|-----------------------------|--------------------|---|-----------------|--|
| routes                      |                    | A parameter that lists multiple HTTP routes. For more information, see <a href="#">Routes [page 649]</a> .  | \${default-uri} | <p>↔ Sample Code</p> <pre>modules: - name: my-app   type: application   parameters:     routes:       - route:         "*foo.my.custom.domain/path"           - route:             foo.my.custom.domain/path       - route:         foo.\$ {default-domain}/path</pre> |
| route-path                  |                    | The context "route-path" which is part of the application default URI. Context path routing is routing based not only on domain names (host header) but also the path specified in the URL.             | n/a             | route-path: /myapp   |
| service-broker-name         |                    | The name of the service broker in the Cloud Foundry environment to be created and registered for the specified application module   | \${app-name}    | <pre>service-broker-name: jobscheduler service-broker-name: \${app-name}</pre>   |
| service-broker-password     |                    | The password used for authentication by the XS controller at the service broker when performing service-related requests. The parameter is mandatory if <code>create-service-broker: true</code> .      |                 | <pre>service-broker- password: \${generated- password}</pre>   |
| service-broker-space-scoped |                    | Makes the service plans of the broker visible only within the targeted space.   | false           | <pre>service-broker-space- scoped: true</pre>  |
| service-broker-url          |                    | Specifies the value of the service broker universal resource locator (URL) to register; service requests are sent to this URL. The parameter is mandatory if <code>create-service-broker: true</code> . |                 | <pre>service-broker-url: \${default-url}</pre>   |

| Parameter              | Read-Only (System) | Description  | Default Value | Example  |
|------------------------|--------------------|--|---------------|--|
| service-broker-user    |                    | The name of the user required for authentication by the XS controller at the service broker when performing service-related requests. The parameter is mandatory if <code>create-service-broker: true</code> . |               | <code>service-broker-user: \${generated-user}</code>   |
| stack                  |                    | Use this parameter to define which pre-built root file system ( <code>rootfs</code> ) you want to use.   | n/a           | <p> <a href="#">Sample Code</a></p> <pre>modules:   - name: foo     type: application     parameters:       stack: cflinuxfs3</pre> |
| stage-timeout          | Write              | Defines how long, in seconds, your application can take during staging before the MTA operation times out.<br><br>See <a href="#">Application-Specific Timeouts [page 636]</a> .                               | 1h            | <pre>modules:   - name: java     .....     parameters:       stage-timeout: 100</pre>  |
| start-timeout          | Write              | Defines how long, in seconds, your application can take to start before the MTA operation times out.<br><br>See <a href="#">Application-Specific Timeouts [page 636]</a> .                                     | 1h            | <pre>modules:   - name: java     .....     parameters:       start-timeout: 100</pre>  |
| task-execution-timeout | Write              | Defines how long, in seconds, your application can take to execute a task before the MTA operation times out.<br><br>See <a href="#">Application-Specific Timeouts [page 636]</a> .                            | 12h           | <pre>modules:   - name: java     .....     parameters:       task-execution-timeout: 100</pre>   |
| tasks                  |                    | Specify tasks, which are available for execution in the current droplet of the application. Also provide use of environment variables which are specified with the <code>env</code> scope.                     | n/a           | <pre>tasks:   - name: task-1     command: some-script.sh     env:       env1: value1       env2: value2</pre>  |
| tcp                    |                    | Specifies whether the application should have TCP type routes.   | false         | <code>tcp:true</code>  |

| Parameter      | Read-Only (System) | Description  | Default Value                              | Example   |
|----------------|--------------------|--|--|---|
| tcps           |                    | Specifies whether the application should have TCPS type routes.  | false                                      | tcps:true   |
| upload-timeout | Write              | Defines how long, in seconds, you can upload your application binary before the MTA operation times out.<br><br>See <a href="#">Application-Specific Timeouts [page 636]</a> . | 1h   | <pre>modules:   - name: java     .....   parameters:     upload-     timeout: 100</pre> |
| timestamp      | Yes                | Current timestamp in milliseconds  | Generated as described in the description. |   |

## Module-Specific Properties

The following properties are supported:

MTA Development and Deployment Module Properties

| Property                | Description   | Default Value | Example   |
|-------------------------|---|---------------|---|
| MTA_WAIT_AFTER_APP_STOP | Adds configurable delay in seconds, after stopping the application. | n/a           | <p>↳ Sample Code</p> <pre>modules: - name: my-app   type: application   properties:     MTA_WAIT_AFTER_APP_STOP: 30</pre> |

## Shared Module Binaries

By default every module has its own binary that is uploaded and deployed in specific manner. It is possible for multiple MTA modules to reference a single deployable binary, for example, an application archive. This means that during deployment, the same application archive is executed separately in multiple applications or application instances, but with different parameters and properties. This results in multiple running applications based on the same source code, which have different configurations and setup. A development project can have one source folder, which is referenced by multiple module entries in the MTA deployment descriptor `mtad.yaml`, as illustrated in the following example:

↳ Sample Code

Multiple MTA Module Entries in the Deployment Descriptor (`mtad.yaml`)

```
_schema-version: "3.1.0"
ID: hello
version: 0.1.0
modules:
  - name: hello-router
```

```

type: java.tomee
path: web/router.war
requires:
- name: backend
  properties:
    backend: ~{url}/content
    name: backend
    url: ~{url}

- name: hello-backend
  type: java.tomee
  path: web/router.war
  provides:
    - name: backend
      properties:
        url: "${default-url}"

```

If deployment is based on an MTA archive, it is not necessary to duplicate the code to have two different deployable modules; the specification for the MTA-module entry in `MANIFEST.MF` is extended, instead. The following (incomplete) example of a `MANIFEST.MF` shows how to use a comma-separated list of module names to associate one set of deployment artifacts with all listed modules:

#### Code Syntax

Multiple MTA Modules Listed in the `MANIFEST.MF` Deployment Manifest

```

Name: web/router.war
MTA-Module: hello-router,hello-backend
Content-Type: application/zip

```

## Related Information

[Managing Service Brokers](#) 

[Parameters and Properties \[page 619\]](#)

[Metadata for Properties and Parameters \[page 630\]](#)

## 4.1.8.1.6.3 Module Hooks

Define and execute hooks at specific phases of module deployment.

#### Note

The module hooks are supported from major schema version 3 onwards. If they are specified but schema version is below 3, they are ignored.

You can use hooks to change the typical deployment process, in this case to enable tasks to be executed during a specific moment of the application deployment. For example, you can set hooks to be executed before or after the actual deployment steps for a module, depending on the applications' need.

When added to the deployment descriptor, module hooks are modeled as follows:

### ↔ Sample Code

```
_schema-version: "3.3"
ID: foo
version: 3.0.0
modules:
  - name: foo
    type: javascript.nodejs
    hooks:
      - name: hook
        type: task
        phases:
          - blue-green.application.before-stop.live
          - blue-green.application.before-stop.idle
    parameters:
      name: foo-task
      command: 'sleep 5m'
```

In the example above, the hook attributes are:

- **name** – defines the name of the hook.
- **type** – defines the type of the hook. Currently, the only supported type is **task**.
- **phases** – defines the specific moment when a hook is executed. The **phases** element denotes that the hook is executed before the application is stopped. The suffixes **live** and **idle** are used during blue-green deployments and indicate when the **before-stop** phase is executed.

### ❖ Example

In the example above, the `blue-green.application.before-stop.idle` phase executes the hook when the new idle applications are redirected to the new live routes, and the `blue-green.application.before-stop.live` is used just before the deletion of the live application.

- **parameters** – defines the parameters of the hook. For the hooks of type **task**, the parameters must define a one-off task.

Depending on the deployment strategy you use, the **phases** values are:

- **For regular deployment**

| Phase  | Supported for types | Description   |
|--|---------------------|---|
| <code>deploy.application.before-stop</code>  | task                | Executed before the application corresponding to the module is stopped. |
| <code>deploy.application.after-stop</code>   | task                | Executed after the application corresponding to the module is stopped.  |
| <code>deploy.application.before-start</code> | task                | Executed before the application corresponding to the module is started. |

- **For blue-green deployment**

| Phase   | Supported for types | Description  |
|---|---------------------|--|
| blue-green.application.before-stop.idle         | task                | Executed before the idle application corresponding to the module is stopped. Idle applications are created as part of blue-green deployments and are restarted (stopped and started again) after the deployment validation phase of their productization. That is after the live routes are mapped to them and their environments are rebuilt to contain only live routes. |
| blue-green.application.before-stop.live         | task                | Executed before the live application corresponding to the module is stopped. Live applications are the ones that are already up and running before the deployment starts.  |
| blue-green.application.after-stop.idle          | task                | Executed after the idle application corresponding to the module is stopped.  |
| blue-green.application.after-stop.live          | task                | Executed after the live application corresponding to the module is stopped.  |
| blue-green.application.before-unmap-routes.live | task                | Executed before unmapping the route of the live application that corresponds to the module.  |
| blue-green.application.before-start.idle        | task                | Executed before the idle application corresponding to the module is started.   |
| blue-green.application.before-start.live        | task                | Executed before the live application corresponding to the module is started.   |

## Module Hooks - Specific Parameters

The table below contains the parameters of the supported module hook types:

Module hooks of type task

| Parameter | Is It Mandatory? | Description  | Default Value         | Example                            |
|-----------|------------------|--|-----------------------|------------------------------------|
| name      | No               | Defines the name of the Cloud Foundry task that should be executed.  | The name of the hook. | name:<br>db_migration              |
| command   | Yes              | Defines the actual command that is executed as a Cloud Foundry task. |                       | command:<br>"bin/rails db:migrate" |

| Parameter  | Is It Mandatory? | Description   | Default Value   | Example                            |
|------------|------------------|---|---|------------------------------------|
| memory     | No               | Defines the memory that is available to the Cloud Foundry task.     | Landscape specific value that is equal to the default application memory.     | memory: 256M<br>memory: 1G         |
| disk-quota | No               | Defines the disk space that is available to the Cloud Foundry task. | Landscape specific value that is equal to the default application disk quota. | disk-quota: 256M<br>disk-quota: 1G |

## Extending Module Hooks Through an Extension Descriptor

You can also extend module hooks through the extension descriptor. To do so, add the code with your specific parameters, similarly to the following example:

### ↔ Sample Code

```
_schema-version: "3.3"
ID: foo-change-command
extends: foo
modules:
  - name: foo
    hooks:
      - name: hook
        parameters:
          command: 'sleep 1m'
```

## Related Information

[MTA Deployment Descriptor Examples \[page 579\]](#)

[Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 578\]](#)

[Defining MTA Extension Descriptors \[page 631\]](#)

[Legacy Blue-Green Deployment \[page 658\]](#)

### 4.1.8.1.6.4 Parameters and Properties

This section contains information about the parameters and properties of a Multitarget Application (MTA).

The values of parameters and properties can be specified at design time, in the MTA development description (`mta.yaml`) or in the MTA deployment descriptor (`mtad.yaml`). In some cases, it is better to declare certain values depending on the deployment, for example, in an extension descriptor file (`myDeployExtension.mtaext`).

The values of parameters and properties might be literals. This way, the result of each deployment will be the same every time. However, by using the placeholders described below, the values might also be set as substitution variables and therefore each deployment might end in a different result depending on the environment or other configurations.

Regardless of the defined value - literal or substitution variable, in most cases the end value for each parameter or property is definitive and well-known before the deployment. This is because the substitution variable values represent a certain template and template resolving is done in the beginning of the MTA deployment.

The module level parameter  `${default-host}` follows the template  `${org}- ${space} - <module_name>`. For example, when it is used in a module `my-module` and the deployment is done within org `my-org`, space `my-space`, the end value of  `${default-host}` will be `my-org-my-space-my-module`.

However, in some cases the parameter or property value is dynamic and cannot be predicted before the deployment. In these cases, the substitution variable is dynamic and is done in a later phase of the MTA deployment.

If you want to learn more about dynamically resolved parameters see the [Dynamic parameters \[page 628\]](#) section below.

The values of properties and parameters are used during the deployment or at runtime of the MTA.

#### ⓘ Note

Both parameters and properties may have literal values, such as strings, integers, etc. This also applies to deeply nested structured values, such as arrays or maps.

#### → Tip

- You can declare metadata for parameters and properties defined in the MTA deployment description; the mapping is based on the parameter or property keys. For example, you can specify if a parameter is **required** (`optional: false`) or can be modified `overwritable: true`.
- Descriptors can contain so-called placeholders (also known as substitution variables), which can be used as sub-strings within property and parameter values. Placeholder names are enclosed by the dollar sign (\$) and curly brackets ({}). For example:  `${host}` and  `${domain}`. For each parameter

"P", there is a corresponding placeholder \${P}. The value of <P> can be defined either by a descriptor used for deployment, or by the deploy service itself. Placeholders can also be used without any corresponding parameters; in this scenario, their value cannot be overridden in a descriptor. Such placeholders are read-only.

## Parameters

Parameters are reserved variables that affect the behavior of the MTA-aware tools, such as the Cloud MTA Build Tool (MBT) or SAP Cloud Deployment service.

Parameters might be used on various levels in the MTA descriptor - top-level, module level, resource level, and dependency level. Based on the parameter applicability it might be used in combination in several places, for example, both on resource and module levels.

Parameters can be "Read-Only" (also known as "System") or "Read-Write" (default value can be overwritten). All parameter values can be referenced as part of other property or parameter value strings. The value of a "Read-Only" parameter cannot be changed in descriptors. Only its value can be referenced using the placeholder notation. To reference a parameter value, use the placeholder notation \${<parameter>}}, for example \${org}

SAP Cloud Deployment service supports a list of parameters and their (default) values:

- [Module-Specific Parameters \[page 601\]](#)
- [Resource-Specific Parameters \[page 591\]](#)
- [Module Hooks - Specific Parameters \[page 617\]](#)
- Generic parameters (table below) that can have the following scopes:
  - Top-level - can be defined on top level.
  - All - can be consumed everywhere throughout the document.

### ⓘ Note

Generic Parameters table contains parameters that might be used on top-level or on all levels. Other supported parameters are distributed in the dedicated pages, for example, module-specific parameters.

The example below shows the parameter `memory` on a module level which defines the amount of memory used by the Cloud Foundry application represented by the module `node-hello-world` during application runtime.

### ↔ Sample Code

```
modules:
  - name: node-hello-world
    type: javascript.nodejs
    path: web/
    parameters:
      memory: 128M
```

- Generic Parameters

| Parameter   | Scope  | Read-Only (System) | Description  | Default Value | Example  |
|---|--------|--------------------|--|---------------|--|
| <pre>apply- namespace: app- names: true/ false</pre><br><pre>service- names: true/ false</pre><br><pre>app- routes: true/ false</pre><br><pre>as- suffix: true/ false</pre> | Global | Write              | <p>Applies namespace to application names (app-names), service names (service-names), application routes (app-routes) as suffix or prefix (as-suffix). If the namespace value is not provided in the CLI options, it is not applied.</p> <p>See <a href="#">Fine-Grained Configuration [page 667]</a>.</p> | true          | <pre>parameters:   apply-   namespace:     app-     names: true     service-     names: true     app-     routes: false     as-     suffix: true</pre> |
|   |        |                    | <p><b> ⓘ Note</b></p> <p>This applies to all applications.</p>   |               |  |
| apps- stage- timeout  | Global | Write              | <p>Defines how long, in seconds, your application can take during staging before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>   | 1h            | <pre>parameters:   apps-stage-   timeout: 100</pre>  |
|   |        |                    | <p><b> ⓘ Note</b></p> <p>This applies to all applications.</p>   |               |  |

| Parameter                   | Scope  | Read-Only (System) | Description   | Default Value                              | Example   |
|-----------------------------|--------|--------------------|---|--|---|
| apps-start-timeout          | Global | Write              | <p>Defines how long, in seconds, your application can take to start before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>          | 1h   | <pre>parameters:   apps-start-   timeout: 100</pre>   |
| apps-task-execution-timeout | Global | Write              | <p>Defines how long, in seconds, your application can take to execute a task before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p> | 12h  | <pre>parameters:   apps-   task-execution-   timeout: 100</pre>                             |
| apps-upload-timeout         | Global | Write              | <p>Defines how long, in seconds, you can upload your application binary before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>      | 1h   | <pre>parameters:   apps-upload-   timeout: 100</pre>  |
| authorization-url           | All    | Yes                | The authorization URL as specified in the cloud controller's /v2/info endpoint.   | Generated as described in the description. | <a href="https://login.cf.sap.hana.ondemand.com">https://login.cf.sap.hana.ondemand.com</a> |
| controller-url              | All    | Yes                | The URL of the cloud controller   | Generated as described in the description. | <a href="https://api.cf.sap.hana.ondemand.com">https://api.cf.sap.hana.ondemand.com</a>     |

| Parameter          | Scope | Read-Only<br>(System) | Description  | Default Value                              | Example                      |
|--------------------|-------|-----------------------|--|--|------------------------------|
| default-domain     | All   | Yes                   | The default domain (configured in the Cloud Foundry environment)   | Generated as described in the description. | accra6024<br>cfapps.acme.com |
| deploy-url         | All   | Yes                   | The deploy service URL for the Cloud Foundry environment   | Generated as described in the description. |                              |
| generated-password | All   | Yes                   | Randomly generated string value that is composed of 16 characters that may contain upper and lower case letters, digits and special characters (_-, @, \$, &, #, *). | Generated as described in the description. | IG@zGg#2g-cvMvsW             |
| generated-user     | All   | Yes                   | A generated user id that is composed of 16 characters that may contain upper and lower case letters, digits and special characters (_-, @, \$, &, #, *).             | Generated as described in the description. | uYi\$d41TzM1-Dm6f            |

| Parameter            | Scope  | Read-Only (System) | Description  | Default Value | Example   |
|----------------------|--------|--------------------|--|---------------|---|
| keep-existing-routes | Global | Write              | <p>When specified on module level, it indicates if the existing routes of the module's corresponding application should be kept even if they are not defined within the deployment and/or extension descriptors.</p> <p>When specified on global level, under the <code>parameters</code> section of the descriptor, it indicates if the existing routes of all applications within that MTA should be kept.</p> | false         | <pre>parameters:   keep-existing-routes: true modules:   - name: foo     type: nodejs     parameters:       keep-existing-routes: false   - name: bar     type: nodejs   - name: baz     type: nodejs</pre> |

#### ⓘ Note

- The module-level variant of the parameter has priority over the global parameter.
- This parameter is typically used when users want to keep the routes they have mapped manually by using the `cf map-route` command. We discourage this approach, as manual operations could lead to inconsistent deployment results and difficult troubleshooting. We recommend you to define all routes in the deployment and/or extension descriptors, which allows for their automatic management.

| Parameter   | Scope | Read-Only<br>(System) | Description  | Default Value                               | Example                              |
|-------------|-------|-----------------------|--|---|--------------------------------------|
| org         | All   | Yes                   | Name of the target organization                              | The current name of the target organization | initial, trial                       |
| protocol    | All   | Yes                   | The protocol used by the Cloud Foundry environment.          | http or https                               | http, https                          |
| space       | All   | Yes                   | Name of the target organizational space                      | Generated as described in the description.  | initial, a007007                     |
| user        | All   | Yes                   | Name of the current user                                     | Generated as described in the description.  |                                      |
| xs-type     | All   | Yes                   | The XS type, Cloud Foundry or XS advanced                    | CF  | CF, XSA                              |
| org-guid    | All   | Yes                   | GUID (Globally Unique Identifier) of the target organization | N/A   | 06564ad5-1b38-458d-8c85-a2e0bcd990a9 |
| space-guid  | All   | Yes                   | GUID (Globally Unique Identifier) of the target space        | N/A   | 06564ad5-1b38-458d-8c85-a2e0bcd990a9 |
| mta-version | All   | Write                 | The version of the MTA                                       | N/A   | 1.0.0                                |
| mta-id      | All   | Write                 | The ID of the MTA  | N/A   | 1.0.0                                |

These parameters can be used with the provides or requires dependencies:

#### Dependency-Specific Parameters

| Parameter    | Scope               | Read-Only<br>(System) | Description  | Default Value | Example  |
|--------------|---------------------|-----------------------|--|---------------|--|
| binding-name | required dependency | Write                 | Provide a binding name for the association between an application and a service instance |               | <pre>modules:   - name: java   .....   requires:     - name:test   parameters:     binding-name: java-test</pre> |

| Parameter       | Scope               | Read-Only (System) | Description   | Default Value                              | Example  |
|-----------------|---------------------|--------------------|---|--|--|
| env-var-name    | required dependency | Write              | Used when consuming an existing service key. Specifies the name of the environment variable that will contain the service key's credentials. See Consumption of existing service keys for more information. | The name of the service key.               | env-var-name: SERVICE_KEY_CREDENTIALS  |
| visibility      | provided dependency | Write              | Specifies the organizations and spaces in which public provided dependencies are visible. See Visibility of cross-MTA configuration for more information.   | In all spaces of the current organization. | <pre>visibility:   - org: foo     space: "*"   - org: bar     space: "*"   - org: baz     space: qux</pre>                                       |
| use-live-routes | provided dependency | Write              | Valid for blue-green deployment. Specify this parameter if you want to provide the routes specified in the descriptor.  | false                                      | <pre>provides:   - name: example-name     properties:       provided-route: \${routes/0/route}     parameters:       use-live-routes: true</pre> |

As an alternative, you can also externalize such configurations in a file. See [Service Creation Parameters \[page 641\]](#).

## Properties

MTA properties are Cloud Foundry application environment variables that are used during application runtime. When an MTA property is set, the SAP Cloud Deployment service injects its key as the environment variable key and its value as the variable value in the application, represented by the corresponding MTA module.

MTA properties can be declared in different levels - module level, resource level, and dependency level.

### Cross-References to Properties

To enable resource properties to resolve values from a property in another resource or module, a resource must declare a dependency. However, these “requires” declarations do not affect the order of the application deployment.

## **Restriction**

It is not possible to reference **list** configuration entries either from resources or “subscription” functionalities (deployment features that are available to subscriber applications).

## **Code Syntax**

Cross-References between Properties in the MTA Deployment Descriptor in a YAML file

```
modules:
  - name: java
    ...
    provides:
      - name: backend
        properties:
          url: ${default-url}/foo
resources:
  - name: example
    type: example-type
    properties:
      example-prop: my-example-prop
  - name: uaa
    type: uaa-type
    requires:
      - name: example
      - name: backend
        properties:
          prop: ~{url}
        parameters:
          param: ~{url}
    properties:
      pro1: ~{example/example-prop}
    parameters:
      config:
        app-router-url: ~{backend/url}
        example-prop: ~{example/example-prop}
```

Note the following about the example above:

- The application name `backend` can depend on the provided content.
- The `java` module provides the `url` property, which can be referenced by other modules. Its value is generated using the default `url` provided to the application during deployment.
- The `java` module property `prop` contains a parameter `url`, which results in an environment variable `<prop>` containing the referenced value.
- References to other properties can also be used for parameter values, for example in values contained in the `provides` section of a module.

## **Fully qualified references to properties**

The SAP Cloud Deployment service supports the extension of the standard syntax for references in module properties. This extension enables you to specify the name of the `requires` section inside the property reference. You can use this syntax extension to declare implicit groups, as illustrated in the following example:

## ↔ Sample Code

Syntax Extension: Alternative Grouping of MTA Properties

```
modules:
  - name: pricing-ui
    type: javascript.nodejs
    properties:
      API: # equivalent to group, but defined in the module properties
        - key: internal
          protocol: ~{price_opt/protocol} #reference to value of protocol
          defined in price_opt of module pricing-backend
        - key: external
          url: ~{competitor_data/url} # reference to string value of property
          'url' in required resource 'competitor_data'
          api_keys: ~{competitor_data/creds} # reference to list value of
          property 'creds' in 'competitor_data'
      requires:
        - name: competitor_data
        - name: price_opt
    - name: pricing-backend
      type: java.tomcat
    provides:
      - name: price_opt
        properties:
          protocol: http ...
    resources:
      - name: competitor_data
        properties:
          url: "https://marketwatch.acme.com/"
          creds:
            app_key: 25892e17-80f6
            secret_key: cd171f7c-560d
```

## Dynamic parameters

In some cases, it is required to use a value that is unknown before deployment. For example, such entity could be the GUID of another service instance. The GUID of a newly created service instance is not known before the creation of said service but it might be consumed during the MTA deployment.

The feature enables an existing Cloud Foundry entity parameter value to be used inside an MTA descriptor.

### ⓘ Note

Currently, the feature is limited to resolving service GUID for the following resource types:

- org.cloudfoundry.managed-service
- org.cloudfoundry.user-provided-service
- org.cloudfoundry.existing-service

Resolved parameters can only be referenced from other resources or from a cross-MTA dependency. If a dynamic parameter is referenced from other places, like modules, the reference won't be resolved.

## ↔ Sample Code

MTA Deployment Descriptor (`mtad.yaml`)

```
_schema-version: 3
ID: sample-dynamic-service-guid
version: 1.0.0
modules:
  - name: sample-app
    type: staticfile
    path: content.zip
    requires:
      - name: test-service
      - name: hana-service
resources:
  - name: test-service
    type: org.cloudfoundry.user-provided-service
    parameters:
      config:
        service-guid-of-db: ~{hana-service/my-db-service-guid}
    requires:
      - name: hana-service
    processed-after: [hana-service]
  - name: hana-service
    type: org.cloudfoundry.managed-service
    parameters:
      service: hana
      service-plan: schema
    properties:
      my-db-service-guid: ${service-guid}
```

## ⓘ Note

In order to use this feature, it is necessary to specify  `${service-guid}` inside the `properties` section of the resource. Consumer of service GUID value must add respective resource name in the `requires` section and specify the `processed-after` parameter because the order of resource processing must be explicitly described.

## Dynamic parameter as a cross-MTA dependency

Resolved parameters can also be referenced from a cross-MTA dependency, allowing the service instance to be referenced from another MTA.

The following example shows the dependency declaration referencing a dynamic parameter in the deployment descriptor of the “provider” MTA:

## ↔ Sample Code

```
_schema-version: 3
ID: dynamic-service-guid-consumer
version: 1.0.0
modules:
  - name: app-consumer
    type: staticfile
    path: content.zip
    requires:
      - name: db-config
        properties:
          reference_instance: ~{db-instanceid}
resources:
  - name: db-config
```

```

type: configuration
parameters:
  provider-id: "dynamic-service-guid-provider:db-guid"
  version: ">=1.0.0"
  target:
    org: ${org}
    space: ${space}

```

The following example shows the dependency declaration in the deployment descriptor of the “consumer” MTA:

#### ↳ Sample Code

```

_schema-version: 3
ID: dynamic-service-guid-consumer
version: 1.0.0
modules:
  - name: app-consumer
    type: staticfile
    path: content.zip
    requires:
      - name: db-config
        properties:
          reference_instance: ~{db-instanceid}
resources:
  - name: db-config
    type: configuration
    parameters:
      provider-id: "dynamic-service-guid-provider:db-guid"
      version: ">=1.0.0"
      target:
        org: ${org}
        space: ${space}

```

For more information about cross-MTA configurations see [Cross-MTA Dependencies \[page 652\]](#).

## 4.1.8.1.6.5 Metadata for Properties and Parameters

It is possible to declare metadata for parameters and properties defined in the MTA deployment description, for example, using the “parameters-metadata:” or “properties-metadata:” keys, respectively; the mapping is based on the keys defined for a parameter or property.

You can specify if a property is required (`optional: false`) or can be modified (`overwritable: true`), as illustrated in the following (incomplete) example:

The `overwritable:` and `optional` keywords are intended for use in extension scenarios, for example, where a value for a parameter or property is supplied at deployment time and declared in a deployment-extension descriptor file (`myMTADeploymentExtension.mtaext`).

You can declare metadata for the parameters and properties that are already defined in the MTA deployment description. However, any parameters or properties defined in the `mtad.yaml` deployment descriptor with the metadata value `overwritable: false` cannot be overwritten by a value supplied from the extension descriptor. In this case, an error would occur in the deployment.

### → Tip

Parameters or properties can be declared as sensitive. Information about properties or parameters flagged as sensitive is not written as plain text in log files; it is masked, for example, using a string of asterisks (\*\*\*\*\*). Note the `secret_token:` element in the example.

### ↔ Code Syntax

Metadata for MTA Deployment Parameters and Properties

```
modules:
  - name: frontend
    type: javascript.nodejs
  parameters:
    memory: 128M
    domain: ${default-domain}
  parameters-metadata:
    memory:
      optional: true
      overwritable: true
    domain:
      overwritable: false
  properties:
    secret_token: ${generated-password}
    backend_types:
      order_management: sap-erp
      data_warehouse: sap-bw
```

## 4.1.8.1.7 Defining MTA Extension Descriptors

The Multitarget Application (MTA) extension descriptor is a YAML file that contains data complementary to the deployment descriptor (from MTA archive). It has a structure similar to the one of the deployment descriptor, and you will normally find it placed outside of the MTA archive.

The data can be environment-specific or deployment-specific, for example, application scaling setup or credentials depending on the user who performs the deployment. This allows you to use the one and the same unmodifiable MTA archive and deploy it with different extension descriptors, resulting in different setups.

When you are deploying an MTA, you can provide several extension descriptors. The descriptors in the chain are considered in order, with each descriptor having a higher priority than the one it extends.

### ⓘ Note

The format and available options within the extension descriptor may change with newer versions of the MTA specification. You must always specify the schema version option when defining an extension descriptor to inform the SAP BTP which MTA specification version should be used. Furthermore, the schema version used within the extension descriptor and the deployment descriptor should always be the same.

### ⓘ Note

Each extension descriptor is defined in a separate file with an extension `.mtaext`.

In the examples below, we have a deployment descriptor, which has already been defined, and several extension descriptors.

MTA deployment descriptor:

### • Example

```
----MTA Deployment descriptor-----
_schema-version: '3.1'
ID: sample.app
version: 0.1.0

modules:
- name: my-app
  type: application
parameters:
  memory: 2GB
```

When an MTA with the above deployment descriptor is being deployed, the following steps are performed:

- Validates the deployment descriptor against the MTA specification version 3.1. If it is not compliant, the deployment will fail at the beginning
- Defines an MTA module with type `application` that will end up into a Cloud Foundry application
- Defines a parameter `memory` that equals to 2GB
- .....
- Creates a Cloud Foundry application that runs with 2GB memory

The following is a basic example of an extension descriptor that adds and overwrites data:

### • Example

```
----prod.mtaext-----
_schema-version: '3.1'
ID: sample.app.prod
extends: sample.app

modules:
- name: my-app
  parameter:
    memory: 4GB
  instances: 2
```

When the same MTA is deployed in combination with the above MTA extension descriptor, using the command `cf deploy <mtar> -e prod.mtaext`, the following steps are performed:

- Validates the deployment descriptor and the extension descriptor against the MTA specification version **3.1**
- Extends the deployment descriptor with an extension descriptor with ID `sample.app.prod` and merges the result:
  - Overwrites the value of the parameter `memory`. It becomes equals to 4GB
  - Adds a new parameter `instances` that equals to 2
- .....
- Creates a Cloud Foundry application which runs on 2 instances with 4GB memory each

In this scenario, the descriptor chain will consist of two descriptors. They will be arranged in this order: `sample.app` followed by `sample.app.prod`.

The following is an example of a second extension descriptor that builds upon the extension descriptor from the previous example:

### • Example

```
----largemtaext-----
_schema-version: '3.1'
ID: sample.app.large
extends: sample.app.prod

modules:
- name: my-app
  parameter:
    instances: 5
    disk-quota: 10GB
```

When you deploy the same MTA in combination with the two MTA extension descriptors, using the command `cf deploy <mтар> -e prod.mtaext,large.mtaext`, the following steps are performed:

- Validates the deployment descriptor and the extension descriptors against the MTA specification version **3.1**
- Extends the deployment descriptor with an extension descriptor with ID `sample.app.prod` and merges the result:
  - Overwrites the value of the parameter `memory`. It becomes equals to 4GB
  - Adds a new parameter `instances` that equals to 2
- Extends the merged descriptor with an extension descriptor with ID `sample.app.large` and merges the result:
  - Overwrites the value of the parameter `instances`. It becomes equals to 5
  - Adds a new parameter `disk-quota` that equals to 10GB
- ...
- Creates a Cloud Foundry application, which runs on 5 instances, each with 4GB of memory and 10GB of disk space

In this scenario, the descriptor chain will consist of three descriptors: `sample.app`, followed by `sample.app.prod`, and then `sample.app.large`.

Extension descriptors can be used to add or override all data types of parameters and properties, following the specified rules.

- Simple data types, such as Boolean, integer and String, are overwritten directly
- Maps (objects) are overwritten and merged in depth

```
----MTA Deployment descriptor-----
ID: sample.postgre

resources:
- name: my-postgre
  type: org.cloudfoundry.managed-service
parameters:
  service-plan: standard
  service: postgresql-db
  config:
    memory: 4
    storage: 200
```

```

----prod-postgre.mtaext-----
ID: sample.postgre.dev
extends: sample.postgre

resources:
  - name: my-postgre
    parameters:
      config:
        storage: 10

```

When using a combination of the MTA deployment descriptor and MTA extension descriptor, the MTA deployment will result in the creation of a service instance with the service offering `postgresql-db`, plan **standard**, running on 4GB memory, and 10GB storage. This example demonstrates how the parameter `config`, a YAML map (JSON Object), is processed. The value from the extension descriptor is merged with the value from the deployment descriptor while retaining the element `memory`. Without using the extension descriptor, the result of the MTA deployment would be a Postgre service instance with 4GB memory and 200GB storage.

Note that lists (arrays) are overwritten by replacing all elements. This means that if you want to preserve any of the elements from the MTA deployment descriptor you need to define them again in the extension descriptor.

```

----MTA Deployment descriptor-----
ID: sample.app

modules:
  - name: my-app
    type: application
    properties:
      my-list: [1,2]

```

```

----app-list.mtaext-----
ID: sample.app.ext
extends: sample.app

modules:
  - name: my-app
    type: application
    properties:
      my-list: [2,3]

```

When you use a combination of the MTA deployment descriptor and an MTA extension descriptor, the resulting MTA deployment will create an application with the environment variable `my-list` set to [2,3]. This example demonstrates how the parameter `my-list`, which is a YAML list (JSON Array), is processed. The value from the extension descriptor completely overwrites the value from the deployment descriptor without retaining element "1". If the extension descriptor is not used, the environment variable `my-list` would be [1,2].

In some cases, there are operation-level parameters (command-line arguments) that take the highest priority over all descriptors. See [Application-Specific Timeouts \[page 636\]](#).

### What is possible to do with an extension descriptor?

You can do the following using an extension descriptor:

- Add new data in properties and parameters on module level, resource level, `provides` and `requires` section level
- Overwrite existing data (in depth) in modules, resources, parameters, properties, `provides` and `requires` sections. This depends on the parameter or property metadata `overwritable`. See [Metadata for Properties and Parameters \[page 630\]](#).

- As of schema version 3.xx, by default parameters and properties are overwritable and optional. If you want to make a certain parameter or property non-overwritable or required, you need to add specific metadata. See [Metadata for Properties and Parameters \[page 630\]](#).

You cannot use an extension descriptor to:

- Add new entities such as modules or resources
- Change module or resource type
- Alter read-only (system) parameters
- Add new provided or required dependencies
- Change the processing order of modules and resources with `deployed-after` and `processed-after` parameters

## Related Information

[Defining MTA Deployment Descriptors for the Neo Environment](#)

[Defining Multitarget Application Archives \[page 576\]](#)

[MTA Module Types, Resource Types, and Parameters for Applications in the Neo Environment](#)

[The Multitarget Application Model v.2](#)

[The Multitarget Application Model v.3](#)

## 4.1.8.1.8 SAP Business Technology Platform Capabilities

This section contains information about how to manage standard Cloud Foundry entities with MTA modelling.

- [Applications \[page 635\]](#)
- [Tasks \[page 639\]](#)
- [Services \[page 641\]](#)
- [Routes \[page 649\]](#)

### 4.1.8.1.8.1 Applications

If you want to create an application in Cloud Foundry and use the SAP Cloud Deployment service, you must create an MTA module first.

In most of cases, an MTA module represents a Cloud Foundry application, but there are cases where they represent an entirely different type of entity, for example content. Modules can have different types and parameters that modify their behavior. See [Modules \[page 595\]](#) for more information.

#### Note

We strongly recommend that you specify the appropriate buildpack for your application. This ensures that the application is handled by the buildpack for which it is intended and tested. It also speeds up application deployment, because only the specified buildpack is downloaded for use during the staging process.

You have to use the `buildpack` module parameter, if you are using any of the following MTA module types:

- `application`
- `custom`
- `javascript.nodejs`

## MTA Deployment Optimizations

The MTA deployment is an incremental process. This means that the state of the artifacts in the Cloud Foundry environment is compared to the desired state described in the `mtad.yaml`. If there's a difference, a set of operations are computed in order to make the MTA reach the desired state. The following optimizations are possible during the MTA redeployment:

- The application bits are uploaded again if the following is true:
  - The application content is changed and there is no cache of the same content for a different application, even in a different space. The Cloud Foundry environment has global caching of application bits without organizational and space isolation.
- The application is restaged and restarted if the following is true:
  - The application attributes, such as commands, buildpacks, memory, `health-check-url`, and so on, are changed.
  - The application environment is changed.
  - There are new services to which the application is bound.
  - There are services the application is bound to, which are discontinued (not available) in the new version of the MTA.

### ⓘ Note

Currently, Cloud Foundry environment applications are restaged if they are bound to any service with parameters. This is necessary, because it's not possible to get the current Cloud Foundry service parameters and compare them with the desired ones.

- The application is bound to service whose configuration parameters have been changed. This requires an update of the service instance and rebind, restage, and restart of the application.
- The service binding parameters are changed. This requires an update of the service binding and restage of the application.
- The MTA version is changed, which requires a change of special application environment variables, managed by the deploy service.

## Application-Specific Timeouts

When creating Cloud Foundry applications as part of the MTA deployment, you have the option to set specific timeouts for different phases of the application's creation or execution. When any of these timeouts occur, the MTA operation fails with a relevant error message to avoid MTA deployment run indefinitely. When any of these timeouts occurs, the MTA operation fails and a relevant error message is displayed. This is to prevent the MTA deployment from running indefinitely.

Currently the following timeouts are supported:

- **Upload timeout** – the time it takes to upload the application binary to the Cloud Controller. Once the upload is complete, the application is ready to use.
- **Stage timeout** – the time it takes to stage the application.
- **Start timeout** – the time it takes to start all application instances.
- **Task execution timeout** – the time between when you run a Cloud Foundry task and when it reaches its final state.

## Priority of Timeout Parameters

You can configure application timeouts at different levels according to your needs. When the same timeout (for example, start timeout) is configured in several places for one MTA deployment, a certain order of priority is applied.

The order provides flexibility when you want to define common timeouts for all applications which are part of an MTA but at the same time, you need to define application-specific timeouts.

The timeout parameters follow the following descending order of priority:

### 1. Operation parameters / Command-Line Options (Highest Priority):

The highest priority is given to operation parameters passed during the MTA deployment. If you provide a timeout value as a command-line option, it overrides any other parameters defined into the MTA descriptor. These operation parameters apply to all applications.

#### ❖ Example

```
cf deploy ... --apps-upload-timeout 40 -apps-upload-timeout 50 --apps-start-timeout 30 -apps-task-execution-timeout 100
```

### 2. Module-Level Parameters:

If you do not provide any command-line arguments, the module-level parameters are used instead.

#### ❖ Example

```
modules:  
  - name: java  
    .....  
  parameters:  
    upload-timeout: 100  
    stage-timeout: 50  
    start-timeout: 60  
    task-execution-timeout:120
```

### 3. Global-Level Parameters:

If neither a command-line option nor a module-level parameter are provided, the global parameters are used. Global parameters are applicable to all applications.

#### ❖ Example

```
parameters:  
  apps-upload-timeout: 50  
  apps-stage-timeout: 60  
  apps-start-timeout: 70  
  apps-task-execution-timeout:110
```

#### 4. Default Values:

If you do not provide any of the above parameters, the application uses the predefined default timeout values. This ensures that there is always a time limit in place to prevent MTA operations from running indefinitely.

Default values:

- Start timeout: 1h
- Upload timeout: 1h
- Stage timeout: 1h
- Task execution timeout: 12h

#### Maximum Allowed Values

- Upload timeout: 3h
- Stage timeout: 3h
- Start timeout: 3h
- Task execution timeout: 24h

### Related Information

[Modules \[page 595\]](#)

## 4.1.8.1.8.2 Docker Images as Part of an MTA Deployment

Deploy Docker images as part of a Multitarget application.

You can deploy your own or 3rd party Docker images in the Cloud Foundry environment by referencing them in an application module.

This type of application deployment is faster, as images are already built. Thus, staging is not required, and also all dependencies are statically included in the image.

#### ⓘ Note

When you reference docker images, MTA packages are not self-sustained - they depend on external repositories hosting images for the deployment.

You can deploy Docker images as MTA modules using the Cloud Foundry command line interface.

To deploy a Docker image as a Cloud Foundry application, your deployment descriptor should reference the image in a module as described in the following example:

#### ↔ Sample Code

```
...
modules:
  name: foo
  type: application
  parameters:
```

```
docker:  
  image: cloudfoundry/test-app  
  username: <username>  
  password: <password>  
  build-parameters: #only required for mta.yaml  
    no-source: true  
...
```

Using the parameters above:

- **image** - you reference the location of the image so that it can be scheduled for download by the platform. You have to provide the location of the image using the format `<registry.domain>:<port>/repo/image:<tag>`. If you don't do so, images are referenced from Docker Hub. See more about deploying an application with Docker in the provided link at the end of the section.
- (Optional) **username** and **password** - you provide credentials in the descriptor only when the image repository requires them.
- **no-source** - set this parameter to `true` only when creating an MTA development descriptor (`mta.yaml`) that is meant to be used with the Cloud MTA Build Tool.

## Related Information

[Cloud Foundry Documentation: Deploying an App with Docker](#)

[CF MTA Examples in GitHub: Deploying Docker Images as CF Apps with an MTA](#)

<https://hub.docker.com/>

[Cloud MTA Build Tool: Configuring a module that does not have source code to build and package](#)

### 4.1.8.1.8.3 Tasks

Create one-off administration tasks or scripts.

During the deployment process, these tasks can be executed against staged applications. The platform creates a new container for them, where they are performed for a specific period until they are completed, after which the container is deleted.

One-off tasks are modeled in accordance to the following structure:

#### Sample Code

```
_schema-version: "3.1"  
ID: foo  
version: 3.0.0  
modules:  
  - name: foo  
    type: javascript.nodejs  
    parameters:  
      no-route: true  
      no-start: true  
      disk-quota: 2GB  
    tasks:  
      - name: example_task  
        command: npm start
```

```
memory: 1GB
```

In the structure above, the parameters stand for the following:

- When the task is triggered, the `npm start` command is executed.
- Entering a specific value in the `memory` parameter is optional. A platform-derived default memory size is used for the execution of the task. You can specify a different value if required.
- The `no-route` parameter specifies if a route is required. Applications whose only purpose is to perform a task and then be stopped usually do not require a route. The default value is `false`.
- Use the `no-start` parameter if you want only the one-off tasks to be executed, that is, without triggering the start of the application. If you do not define this parameter, tasks are automatically executed after the applications are started.

### Note

The values for application memory and task memory do not have a dependency. This is also valid for the allowed disk quota.

### Tip

For more information about one-off tasks, see [MTA Deployment Descriptor Examples \[page 579\]](#).

The following codeblock contains an example for a database migration task:

### Sample Code

```
_schema-version: "3.1"
ID: foo
version: 3.0.0
modules:
  - name: foo
    type: javascript.nodejs
parameters:
  no-route: true
  no-start: true
  memory: 1GB
  disk-quota: 2GB
tasks:
  - name: db_migration
    command: "bin/rails db:migrate"
    memory: 256M          #This parameter is optional.
    disk-quota: 128M      #This parameter is optional.
```

## Related Information

[Using Tasks](#) 

## 4.1.8.1.8.4 Services

If you want to create a service in the Cloud Foundry environment using the SAP Cloud Deployment service, you must define an MTA resource first. These resources can have different types and parameters which modify their behavior. There is a predefined set of supported resource types, that in most cases represents a certain combination of service offering and plan. See [Predefined MTA Resource Types \[page 588\]](#).

### ↔ Sample Code

```
resources:  
  - name: my-xsuaa-service  
    type: com.sap.xs.uaa
```

The result will be a service instance called `my-xsuaa-service` with service offering `xsuaa` and service plan `application`.

For a more flexible approach, see [Generic MTA Resource Types \[page 587\]](#) and [Resource-Specific Parameters \[page 591\]](#).

### ⓘ Note

In most of cases, MTA resources represent a Cloud Foundry service, but there are cases where they represent a different platform entity, for example a service key. They can even serve only to a group of configurations. See [Resources \[page 585\]](#) for more information.

## 4.1.8.1.8.4.1 Service Creation Parameters

Some services support additional configuration parameters in the `create-service` request. These parameters are parsed in a valid JSON object containing the service-specific configuration parameters.

The deploy service supports the following methods for the specification of service creation parameters:

- Method A - an entry in the MTA deployment descriptor or the extension
- Method B - a JSON file with the required service configuration parameters

### ⓘ Note

If service-creation information is supplied both in the deployment (or extension) descriptor **and** in a supporting JSON file, the parameters specified directly in the deployment (or extension) descriptor override the parameters specified in the JSON file.

## Service creation parameters - method comparison

| Method A   | Method B  | Combination of the two methods   |
|--|---|--|
| <p>↔ Sample Code</p> <pre>MTA deployment descriptor or extension  resources:   - name: java-uaa     type: com.sap.xs.uaa     parameters:       config:         xsappname: java-hello-world</pre> | <p>↔ Sample Code</p> <pre>MTA deployment descriptor or extension  resources:   - name: java-uaa     type: com.sap.xs.uaa</pre>                | <p>↔ Sample Code</p> <pre>resources:   - name: java-uaa     type: com.sap.xs.uaa     parameters:       path: &lt;path to directory&gt;       config:         xsappname: java-hello-world</pre> |
|  | <p>↔ Sample Code</p> <pre>xs-security.json  {   "xsappname": "java-hello-world" }</pre>   | <p>↔ Sample Code</p> <pre>xs-security.json  {   "xsappname": "java-hello-world" }</pre>  |
|  | <p>↔ Sample Code</p> <pre>Additional entry in MANIFEST.MF  Name: xs-security.json MTA-Resource: java-uaa Content-Type: application/json</pre> | <p>↔ Sample Code</p> <pre>Additional entry in MANIFEST.MF  Name: xs-security.json MTA-Resource: java-uaa Content-Type: application/json</pre>  |

Using method A, all parameters under the special `config` parameter are used for the service creation request. This parameter is optional.

### → Tip

You can use the optional `path` parameter to navigate to a JSON file containing service-specific parameters and configurations. The content of the file is used for the creation or update of the service instance.

Note that if you are deploying from a local directory using an `mtad.yaml`, it needs to contain the path to the JSON file, so that the relevant `MANIFEST.MF` entry is generated.

Using method B, there are dependencies on further configuration entries in other configuration files. For example, if you use this JSON method, an additional entry must be included in the `MANIFEST.MF` file which defines the path to the JSON file containing the parameters as well as the name of the resource for which the parameters should be used.

## Related Information

[Standalone Application Router - HTML5 Application Repository](#) ↗

[Managing Service Instances with the cf CLI](#) ↗

### 4.1.8.1.8.4.2 Service Binding Parameters

Some services support additional configuration parameters with the bind request. These parameters are passed in a valid JSON object containing service-specific configuration parameters, provided either in-line or in a file. For a list of supported configuration parameters, see the documentation for the particular service offering.

#### ⓘ Note

At the time being user-provided services do not support arbitrary binding parameters.

The SAP Cloud Deployment service supports the following methods for the specification of service-binding parameters:

- Method 1 - In-line parameters into MTA descriptors (deployment or extension descriptors)
- Method 2 - JSON file with external configuration, containing the required service-configuration parameters
- A combination of the first two methods, as described in the table below

#### ⓘ Note

If service-binding information is supplied both in the MTA's deployment (or extension) descriptor **and** in a supporting JSON file, the parameters specified directly in the deployment (or extension) descriptor override the parameters specified in the JSON file.

In the MTA deployment descriptor, the `requires` dependency between a module and a resource represents the binding between the corresponding application and the service created from them (if the service has a `type`). For this reason, the `config` parameter is nested in the `requires` dependency parameters, and a distinction must be made between the `config` parameter in the `modules` section and the `config` parameter used in the `resources` section (for example, when used for service-creation parameters).

## Service Binding Parameters - method comparison

| Method 1   | Method 2  | Combination of the two methods   |
|--|---|--|
| <p>↔ Sample Code</p> <pre>MTA deployment descriptor or extension  modules:   - name: node-hello-     world-backend     type:     javascript.nodejs     requires:       - name: node-         hdi-container     parameters:       config:</pre> <p>permissions:<br/>debugging</p> | <p>↔ Sample Code</p> <pre>A JSON file with the required service- configuration parameters  modules:   - name: node-hello-     world-backend     type:     javascript.nodejs     requires:       - name: node-         hdi-container</pre> | <p>↔ Sample Code</p> <pre>modules:   - name: node-hello-     world-backend     type:     javascript.nodejs     requires:       - name: node-         hdi-container     parameters:       config:</pre> <p>permissions:<br/>debugging</p> |
|  | <p>↔ Sample Code</p> <pre>Content of xs-hdi.json  {   "permissions": "debugging" }</pre>  | <p>↔ Sample Code</p> <pre>Content of xs-hdi.json  {   "permissions": "debugging" }</pre>   |
|  | <p>↔ Sample Code</p> <pre>Additional entry in MANIFEST.MF  Name: xs-hdi.json MTA-Requires: node- hello-world-backend/ node-hdi-container Content-Type: application/json</pre>   | <p>↔ Sample Code</p> <pre>Additional entry in MANIFEST.MF  Name: xs-hdi.json MTA-Requires: node- hello-world-backend/ node-hdi-container Content-Type: application/json</pre>  |

### ⓘ Note

The examples are only illustrative for service-binding parameters. Each service offering supports its own set of parameters.

Method 1 shows how to define the service-binding parameters in the MTA deployment descriptor (`mtad.yaml`). If you use this method, all parameters under the special `config` parameter are used for the service-bind request. This parameter is optional.

Method 2 shows how to define the service-binding parameters for a service-bind request in a JSON file. Using this method, there are dependencies on entries in other configuration files. For example, when using the JSON method, an additional entry must be included in the `MANIFEST.MF` file. This new entry has to define the path to the JSON file that contains the parameters and the name of the resource for which the parameters should be used.

Note that when you use the combination of the two methods, the parameters defined in the descriptor have higher priority than the ones defined in the JSON file.

#### ⓘ Note

To avoid ambiguities, the name of the module is added as a prefix to the name of the `requires` dependency; the name of the manifest attribute uses the following format: `<module-name>#<requires-dependency-name>`.

### 4.1.8.1.8.4.3 Service Tags

Some services provide a list of tags that are later added to the `<VCAP_SERVICES>` environment variable. These tags provide a more generic way for applications to parse `<VCAP_SERVICES>` for credentials.

You can also provide custom tags when creating a service instance. To inform the SAP Cloud Deployment service about custom tags, you can use the special `service-tags` parameter. This parameter must be located in the `resources` definition that represent the managed services, as illustrated in the following example:

#### ↗ Sample Code

Defining Service Tags in the MTA Deployment Descriptor

```
resources:  
  - name: nodejs-uaa  
    type: com.sap.xs.uaa  
    parameters:  
      service-tags: [ "custom-tag-A", "custom-tag-B" ]
```

#### ⓘ Note

Some service tags are inserted by default, for example, `xsuaa`, for the XS User Account and Authentication (UAA) service.

### 4.1.8.1.8.4.4 Service Broker Creation

Service brokers are applications that advertise a catalog of service offerings and service plans, as well as interpreting calls for creation, binding, unbinding, and deletion. The deploy service supports automatic creation (and update) of service brokers as part of an application deployment process.

An application can declare that a service broker should be created as part of its deployment process, by using the following parameters in its corresponding module in the MTA deployment (or extension) descriptor:

#### → Tip

You can use placeholders  `${ }`  in the service-URL declaration.

## ↔ Sample Code

```
- name: jobscheduler-broker
  properties:
    user: ${generated-user}
    password: ${generated-password}
  parameters:
    create-service-broker: true
    service-broker-space-scoped: true
    service-broker-name: jobscheduler
    service-broker-user: ${generated-user}
    service-broker-password: ${generated-password}
    service-broker-url: ${default-url}
```

### ⓘ Note

By default, defined service brokers in MTA are attempted to be registered as standard service broker which is available in all orgs and spaces. In practice, however, this is forbidden, as the registration requires CF admin access, which is not provided to anyone.

For test purposes, space-scoped brokers can be registered. In order to do that, parameter `service-broker-space-scoped` needs to be equal to `true`.

The value of the “ `${generated-user}` ” and “ `${generated-password}` ” placeholders in the `properties` section is the same as in the `parameters` section. The service-broker application uses this mechanism to inject the user-password credentials.

The `create-service-broker` parameter should be set to `true` if a service broker must be created for the specified application module. You can specify the name of the service broker with the `service-broker-name` parameter; the default value is  `${app-name}` . The `service-broker-user` and `service-broker-password` are the credentials that will be used by the controller to authenticate itself to the service broker in order to make requests for creation, binding, unbinding and deletion of services. The `service-broker-url` parameter specifies the URL on which the controller will send these requests.

### ⓘ Note

During the creation of the service broker, the Cloud Controller makes a call to the service-broker API to inquire about the services and plans the service broker provides. For this reason, an application that declares itself as a service broker must implement the service-broker application-programming interface (API). Failure to do so might cause the deployment process to fail.

### ⓘ Note

Normally, the registration of a space-scoped broker is successful, because it requires SpaceDeveloper privileges of the user. However, for a global registration of the service broker, global admin privileges are needed, which the platform developer usually does not have. In such cases, the MTA deployment would fail. To solve this, do not use the `--do-not-fail-on-missing-permissions` option, which will result in skipping the step with a warning.

## 4.1.8.1.8.4.5 Service Keys

The consumption of existing service keys from applications is an alternative of service bindings. The application can use the service key credentials and consume the service.

### Creation and Update of Service Keys

A `service-keys` parameter can be created or updated when a service instance is being created or updated. It has to be defined under the resources that represent services which support service keys.

#### ↔ Sample Code

```
resources:
- name: my-service-instance
  type: org.cloudfoundry.managed-service
  parameters:
    service-keys: # Specifies the service keys that should be created for the
      respective service instance. Optional parameter.
      - name: tool-access-key # Specified the service key name. Mandatory
        element.
        config: # Specified the service key configuration. All entries under
          this map element are used for the service key creation request. Optional
          element.
          permissions: read-write
      - name: reader-endpoint
        config:
          permissions: read-only
```

As shown in the example above, every service key entry under the `service-keys` parameter supports optional configuration parameters that can be defined under the `config` parameter.

### Consumption of Service Keys

To be consumed, the existing service keys are modeled as a resource of type `org.cloudfoundry.existing-service-key`. MTA modules might be set to depend on these resources by using a configuration in the `requires` section, which results in an injection of the service key credentials in the application environment.

#### ↔ Sample Code

```
modules:
- name: app123
  type: javascript.nodejs
  requires:
    - name: service-key-1
      parameters:
        env-var-name: keycredentials
...
resources:
- name: service-key-1
  type: org.cloudfoundry.existing-service-key
  parameters:
    service-name: service-a
```

```
service-key-name: test-key-1
```

As a result, the application `app123` has the environment variable `keycredentials`, with value the credentials of the existing service key `test-key-1` of service `service-a`.

### ⓘ Note

Note that only the parameter `service-name` is mandatory. It defines which service is used by the application.

The following parameters are optional:

- `service-key-name-` resource parameter, which defines which service key of the defined service is used. The default value is the name of the resource.
- `env-var-name` - required dependency parameter, which defines what is the name of the new environment variable of the application. The default value is the service key name.

## Automatic Service Key Renewal

It's a common use case to want an automatic service key renewal (e.g. if you want to rotate generated service credentials or certificates).

Service key renewal on MTA redeployment is supported by using changed key names. You can do this either by manually changing the service key name in the descriptor or by using a dynamic parameter as part of it -  `${timestamp}`. This parameter will always resolve to the current timestamp at the time of deployment. This means that the key will have a different name on each subsequent deployment. This ensures that it will be considered as new and created. Afterwards the old key will be deleted. This functionality applies to service keys modelled under a resource using the `service-keys:` parameter. Example descriptor for automated key renewal/rotation:

### ↗ Sample Code

```
...
resources:
  - name: my-service
    type: org.cloudfoundry.managed-service
    parameters:
      service-plan: plan
      service: offering
      service-keys:
        - name: my-rotating-key-${timestamp}
...

```

To consume a service key with a changing name in an application environment, make sure to use the parameter `env-var-name` when defining the service key `requires` section in the consuming module. This will ensure that the service key details are persisted under a static alias in the application environment. e.g.

### ↗ Sample Code

```
modules:
  - name: my-app
...

```

```

requires:
  - name: rotating-key
    parameters:
      env-var-name: *keycredentials*
...
resources:
  - name: my-service
    type: org.cloudfoundry.managed-service
    parameters:
      service-keys:
        - name: rotating-key-${timestamp}
  - name: rotating-key
    type: org.cloudfoundry.existing-service-key
    parameters:
      service-name: my-service
      service-key-name: rotating-key-${timestamp}

```

## 4.1.8.1.8.5 Routes

This section describes how developers or administrators have to configure application routes using the MTA modelling.

Routes are defined on module level by using the `routes` parameter. The parameter accepts list of one or many HTTP routes. It is a combination of the old parameters `host`, `domain`, `port` and `route-path`, which encompasses the full addresses to which to bind a module.

In case the new `routes` parameter and the old ones are available, the `routes` value is used and the values of the old parameters are ignored. Each route for the application is created if it does not already exist.

### ⓘ Note

- The `routes` parameter is a list of entries, where every entry is a map of key-value pairs describing a route. Key `route` is required - this is the Cloud Foundry route (cf route) to be created and bound to the application. Multiple optional keys are supported, including `apply-namespace` and `no-hostname`.
- The deployer reads the `route` entry in a way consistent with cf App Manifest files. It interprets all included data up to the first full-stop ' .' as the host, then everything up the forward-slash ' / ' as the domain, and the rest (if present) as the path.

### → Remember

Note the following:

- if you do not define the route using the `routes` parameter or the `host` and `domain` parameters, a placeholder `default-uri` is automatically applied. This placeholder is resolved to  `${default-host} . ${default-domain}`. The `default-host`, as part of the URI, is resolved to  `${org} - ${space} - <module_name>`. For more information about `default-` parameters, see [Module-Specific Parameters \[page 601\]](#).
- We recommend that you explicitly set the `routes` parameter instead of using the automatically set default route.

## ⌚ Example

In order to reference a specific given route inside a deployment descriptor (for example for a `provides` section), use the following syntax, which provides the first given route:

```
provides:  
  - name: foo  
    properties:  
      url: "${protocol}://${routes/0/route}"
```

If you want to create a route with a wildcard hostname, use an asterisk in quotes ("\*").

```
modules:  
- name: my-app  
  type: application  
  parameters:  
    routes:  
      - route: "*foo.my.custom.domain/path"  
      - route: "foo.my.custom.domain/path"  
      - route: "foo.${default-domain}/path"
```

## Deploy applications to a route without a host name

To bind an app to a route without a host name you can use the `routes` and `no-hostname` parameters. Such a route is a subdomain of an existing domain with no host attached at the front. Note that `no-hostname` is a boolean parameter, and must follow every route definition. For example, if you have 2 or more routes without host names in your MTA then you need to add that parameter under each of those routes. If for a given route the `no-hostname` parameter is missing, its value defaults to `false`.

## leftrightarrow Sample Code

```
routes:  
  - route: host1.some-domain.com  
  - route: host2.some-domain.com  
    no-hostname: false  
  - route: subdomain.some-domain.com  
    no-hostname: true
```

In the example above, `no-hostname: false` has no effect on the second route, since that is the default value, while `host2` is registered as a host to the route. The `no-hostname: true` parameter, however conveys to the deployer that `subdomain` is not a separate host, and the whole route is used as a Cloud Foundry domain - subdomain of `some-domain.com`, if present.

## ⚠ Caution

The MTA deployer attempts to create the subdomain if it does not exist, but cannot resolve any specific corner cases about subdomain/domain ownership permissions. If the subdomain has already been created, the deployer maps the application to it.

## Deploy applications to http2 routes

If you want to enable http2 routing traffic to MTA modules (applications) you can use the `protocol: http2` parameter on route level.

Note that the protocol value must be `http1` or `http2` and must be included in every route in order for it to take effect. If the `protocol` parameter is not specified, the value will be set to `http1` by default.

### ↔ Sample Code

```
routes:  
  - route: http1-route.some-domain.com  
  - route: http1- route.some-domain.com  
    protocol: http1 # The default value if not provided  
  - route: http2- route.some-domain.com  
    protocol: http2
```

## Deploy applications to a route with a namespace

To bind an application to a route with a namespace, use the `routes` and `apply-namespace` parameters. When the value of the `apply-namespace` parameter is set to `true`, a prefix with the namespace value will be added to the route.

### ↔ Sample Code

```
routes:  
  - route: host1.some-domain.com  
    apply-namespace: true  
  - route: host2.some-domain.com  
    apply-namespace: false
```

If you don't explicitly set the `apply-namespace` parameter for a specific route, its value will be determined by the application's `apply-namespace` parameter. This means that if the application has a namespace defined, that namespace will also be applied to the bound route. Conversely, if the application does not have a namespace, the route will not have one either.

For more information, see [Fine-Grained Configuration \[page 667\]](#).

### 4.1.8.1.9 Features

The section describes multitarget application-specific features available in the Cloud Foundry environment of the SAP Business Technology Platform.

Each feature has its own productive benefit. For example, the blue-green deployment enables the zero-downtime update of your multitarget application.

- [Cross-MTA Dependencies \[page 652\]](#)
- [Blue-Green Deployment of Multitarget Applications \[page 657\]](#)

- [Legacy Blue-Green Deployment \[page 658\]](#)
- [Blue-Green Deployment Strategy \[page 660\]](#)
- [\(Experimental\) Namespaces \[page 666\]](#)
- [Parallel Module Deployment \[page 672\]](#)
- [Sequential Resource Processing \[page 674\]](#)
- [Features Related to SAP Alert Notificaton service for SAP BTP \[page 675\]](#)
- [Deploying Content with Generic Application Content Deployment \[page 677\]](#)
- [Docker Images as Part of an MTA Deployment \[page 638\]](#)

### 4.1.8.1.9.1 Cross-MTA Dependencies

In addition to having dependencies to one or multiple modules or resources in the same MTA, modules can have dependencies with modules from other MTAs as well. For these so-called cross-MTA dependencies to work, the MTA that **provides** the dependencies must be set to declare them as “public”. Use the `configuration` resource to declare when one module consumes configurations from other MTAs.

#### ⚠ Caution

Do not use this feature for storing sensitive content, for example credentials, as this data is stored unencrypted.

#### ⓘ Note

The declaration method requires adding a resource in the deployment descriptor; the additional resource defines the provided dependency from the other MTA.

### Cross-MTA Dependency Method: Resource Type “`configuration`”

This method can be used to access any entry that is present in the configuration registry. The parameters used in this cross-MTA declaration method are `provider-nid`, `provider-id`, `version`, and `target`. The parameters are all optional and are used to filter the entries in the configuration registry based on their respective fields. If any of these parameters is not present, the entries will not be filtered based on their value for that field. The `version` parameter can accept any valid Semver ranges.

When used for cross-MTA dependency resolution the `provider-nid` is always “`mta`”, the `provider-id` follows the format `<mta-id>:<mta-provides-dependency-name>` and the `version` parameter is the version of the provider MTA. In addition, as illustrated in the following example, the `target` parameter is structured and contains the name of the organization and space in which the provider MTA is deployed. In the following example, the placeholders  `${org}`  and  `${space}`  are used, which are resolved to the name of the organization and space of the consumer MTA. In this way, the provider MTA is deployed in the same space as the consumer MTA.

#### ⓘ Note

As of version 3.0 of the MTA specification, the provided dependencies are no longer public by default. They have to be explicitly declared public for cross-MTA dependencies to work.

The following example shows the dependency declaration in the deployment descriptor of the “consumer” MTA :

#### ↔ Sample Code

Consumer MTA Deployment Descriptor (`mtad.yaml`)

```
_schema-version: "3.1"
ID: com.sap.sample.mta.consumer
version: 0.1.0
modules:
  - name: consumer
    type: java.tomee
    requires:
      - name: message-provider
        properties:
          message: ~{message}
resources:
  - name: message-provider
    type: configuration
    parameters:
      provider-nid: mta
      provider-id: com.sap.sample.mta.provider:message-provider
      version: ">=1.0.0"
      target:
        org: ${org}      # Specifies the org of the provider MTA
        space: ${space}
```

#### → Tip

If no target organization or space is specified by the consumer, then the current organization and space are used to deploy the provider MTA.

The following example shows the dependency declaration in the deployment descriptor of the “provider” MTA :

#### ↔ Sample Code

Provider MTA Deployment Descriptor (`mtad.yaml`)

```
_schema-version: "3.1"
ID: com.sap.sample.mta.provider
version: 2.3.0
modules:
  - name: provider
    type: javascript.nodejs
    provides:
      - name: message-provider
        public: true
        properties:
          message: "Hello! This is a message provided by application \">${app-name}\", deployed in org \">${org}\\" and space \${space}\\"!"
```

## Cross-MTA Configuration Visibility

A “consumer” module must explicitly declare the organizations and spaces in which a “provider” is expected to be deployed, except if it is the same space as the consumer. The “provider” can define a white list that specifies

the organizations and spaces from which the consumption of configuration data is permitted. It is not required to white list all the provider's spaces in its own organization.

### ⓘ Note

Previously, registry entries were visible from all organizations by default. Now, the default visibility setting is "visible within the current organization and all the organization's spaces".

White lists can be defined on various levels. For example, a visibility white list could be used to ensure that a provider's configuration data is visible in the local space only, in all organizations and spaces, in a defined list of organizations, or in a specific list of organization and space combinations.

The options for white lists and the visibility of configuration data are similar to the options available for managed services. However, for visibility white lists, space developers are authorized to extend the visibility of configuration data beyond the space in which they work, without the involvement of an administrator. An administrator is required to release service plans to certain organizations.

Visibility is declared on the provider side by setting the parameter `visibility:` (of type 'sequence'), containing entries for the specified organization (`org:`) and space (`space:`). If no `visibility:` parameter is set, the default visibility value `org: ${org}, space: '*'` is used, which restricts visibility to consumers deployed into all spaces of the provider's organization. Alternatively, the value `org: '*'` can be set, which allows to bindings from all organizations and spaces. The white list can contain entries at the organization level only. This, however, releases configuration data for consumption from all spaces within the specified organizations, as illustrated in the following (annotated) example.

### → Tip

Since applications deployed in the same space are always considered "friends", visibility of configuration data in the local space is always preserved, no matter which visibility conditions are set.

### ↗ Sample Code

```
provides:
  - name: backend
    public: true
  parameters:
    visibility:          # a list of possible settings:
      - org: ${org}       # for local org
        space: ${space}   # and local space
      - org: org1         # for all spaces in org1
      - org: org2         # for the specified combination (org2,space2)
        space: space2
      - org: ${org}         # default: all spaces in local org
      - org: '*'           # all orgs and spaces
      - org: '*'           # every space3 in every org
        space: space3
```

The authorization model ensures the following rules apply:

- Only those users in the white-listed spaces can read or consume the provided configuration data.
- Only users with the role "SpaceDeveloper" in the configuration-data provider's space can modify (edit or delete) configuration data.

## 4.1.8.1.9.1.1 Plug-ins

The SAP Cloud Deployment service supports a method that allows a multitarget application to consume multiple configuration entries for each `requires` dependency.

The following is an example for multiple `requires` dependencies in the MTA Deployment Descriptor (`mtad.yaml`):

### → Sample Code

```
_schema-version: "2.1"
ID: com.acme.framework
version: "1.0" modules:
  - name: framework
    type: javascript.nodejs
    requires:
      - name: plugins
        list: plugin_configs
        properties:
          plugin_name: ~{name}
          plugin_url: ~{url}/sources
        parameters:
          managed: true # true | false. Default is false
resources:
  - name: plugins
    type: configuration
    parameters:
      target:
        org: ${org}
        space: ${space}
      filter:
        type: com.acme.plugin
```

The MTA deployment descriptor shown in the example above contains a module that specifies a `requires` dependency to a configuration resource. Since the `requires` dependency has a `list` property, the deployment service attempts to find multiple configuration entries that match the criteria specified in the configuration resource.

### → Tip

It is possible to create a subscription for a **single** configuration entry, for example, where no “`list:`” element is defined in the required dependency.

### ① Note

The `filter` parameter can be used in combination with other configuration resource specific parameters, for example: `provider-nid`, `provider-id`, `target`, and `version`.

The resource itself contains a `filter` parameter that is used to filter entries from the configuration registry based on their content. In the example shown above, the filter only matches entries that are provided by an MTA deployed in the current space, which have a `type` property in their content with a value of `com.acme.plugin`.

If the `list` element is missing, the values matched by the resources filter are **single** configuration entries – not the usual list of multiple configuration entries. In addition, if either no value or multiple values are found during the deployment of the consuming (subscribing) MTA, the deployment operation fails. If a provider (plug-in) adds more configuration details after the deployment of subscriber applications, these applications won't get

the new information right away. They will only receive these details when they get updated. However, this update will not be successful because there will be several configuration entries available at that point.

The XML document in the following example shows some sample configuration entries, which would be matched by the filter if they were present in the registry.

#### ↔ Sample Code

##### MTA Configuration Entries Matched in the Registry

```
<configuration-entry>
  <id>8</id>
  <provider-nid>mta</provider-nid>
  <provider-id>com.sap.sample.mta.plugin-1:plugin-1</provider-id>
  <provider-version>0.1.0</provider-version>
  <target-space>2172121c-1d32-441b-b7e2-53ae30947ad5</target-space>
  <content>{ "name": "plugin-1", "type": "com.acme.plugin", "url": "https://xxx.mo.sap.corp:51008" }</content>
</configuration-entry>
<configuration-entry>
  <id>10</id>
  <provider-nid>mta</provider-nid>
  <provider-id>com.sap.sample.mta.plugin-2:plugin-2</provider-id>
  <provider-version>0.1.0</provider-version>
  <target-space>2172121c-1d32-441b-b7e2-53ae30947ad5</target-space>
  <content>{ "name": "plugin-2", "type": "com.acme.plugin" }</content>
</configuration-entry>
```

The JSON document in the following example shows the environment variable that is created from the `requires` dependency defined in the example deployment descriptor above, assuming that the two configuration entries shown in the XML document were matched by the filter specified in the configuration resource.

#### ⓘ Note

References to non-existing configuration entry content properties are resolved to “`null`”. In the example above, the configuration entry published for `plugin-2` does not contain a `url` property in its content. As a result, the environment variable created from that entry is set to “`null`” for `plugin_url`.

#### ↔ Sample Code

##### Application Environment Variable

```
plugin_configs: [
  {
    "plugin_name": "plugin-1",
    "plugin_url": "https://xxx.mo.sap.corp:51008/sources"
  },
  {
    "plugin_name": "plugin-2",
    "plugin_url": null
  }
]
```

Requires dependencies support a special parameter named “`managed`”, which registers as a “subscriber” the application created from the module containing the `requires` dependency. This registration causes the application’s environment to be updated if any new configuration entries are published in the configuration registry during the deployment of another MTA. Those new entries must also match the subscription filter of the application. This will cause the application to restart so it can recognize the updated environment state.

#### → Tip

When starting the deployment of an MTA (with the `cf deploy` command), you can use the special option `--no-restart-subscribed-apps` to specify that, if the publishing of configuration entries created for that MTA result in the update of a subscribing application's environment, then that application should **not** be restarted.

### 4.1.8.1.9.2 Blue-Green Deployment of Multitarget Applications

Use the blue-green deployment technique by running two identical production environments, allowing seamless updates without downtime for Cloud Foundry Multitarget applications.

#### ⚠ Restriction

Blue-green deployment is supported only for Cloud Foundry applications. It is not supported for bound services, such as service instances and their configuration, workflow content, and HTML5 repository content, among others. Live and idle applications are bound to the same service instances.

By using the blue-green deployment technique, you can update the system without downtime and with reduced risk. During the process, you can perform tests and validation of the new application version using productive data. A classical blue-green deployment of stateless applications, which are modeled as an MTA, is supported.

In the context of Multitarget applications, you have the following options for using blue-green deployment:

- [Legacy Blue-Green Deployment \[page 658\]](#) - where the production environments are called "blue" and "green"
- [Blue-Green Deployment Strategy \[page 660\]](#) - where the production environments are called "live" and "idle"
- [\(Experimental\) Incremental Blue-Green Deployment Strategy \[page 663\]](#) - an alternative to the standard blue-green deployment strategy, where the "idle" application is incrementally scaled up to the desired instance count while the "live" application instances are gradually stopped

### Related Information

[Using Blue-Green Deployment To Reduce Downtime](#) ↗

## 4.1.8.1.9.2.1 Legacy Blue-Green Deployment

Use the legacy blue-green deployment strategy of Multitarget applications.

### Prerequisites

#### ⚠ Restriction

Blue-green deployment is supported only for Cloud Foundry applications. It is not supported for bound services, such as service instances and their configuration, workflow content, and HTML5 repository content, among others. Live and idle applications are bound to the same service instances.

- You have a previously deployed MTA, with functional productive applications and routes:



- You have ensured that the changes in the new version of the application are compatible with the old version. This includes changes in the database tables, UAA configurations, and so on.

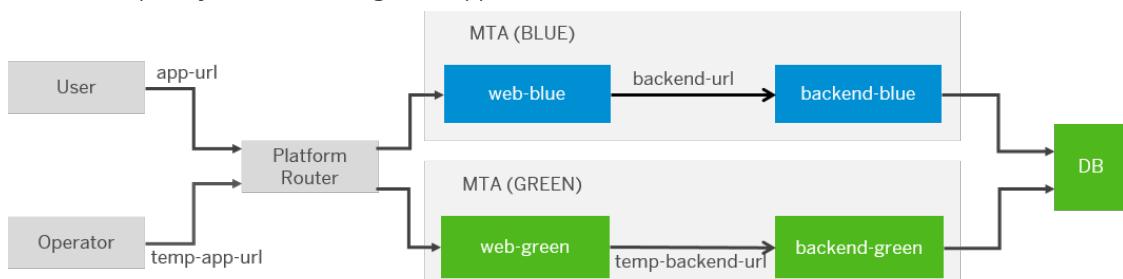
### Procedure

1. Deploy your updated MTA (the green version) by executing the `cf bg-deploy <your-mta-archive-v2>` command.

The first action is that all MTA services are updated.

This action:

- creates new applications adding “green” to their existing application names
- creates temporary routes to the “green” applications.



- interrupts the process showing a message similar to the following:

## ↔ Output Code

```
Process needs additional user input  
Use "cf bg-deploy -i 469520 -a resume" to resume the process  
Use "cf bg-deploy -i 469520 -a abort" to abort the process
```

2. Optionally, test the “green” version using the temporary routes.

### ⓘ Note

You can skip this step by using one of the following command line options:

- **--no-confirm** - you have to use it when starting the process.
- **--skip-idle-start** - this option will also skip the start of the newly deployed applications on temporary routes.

3. If you do not want to make the “green” version available, abort the process using `cf deploy -i <operation ID> -a abort`.

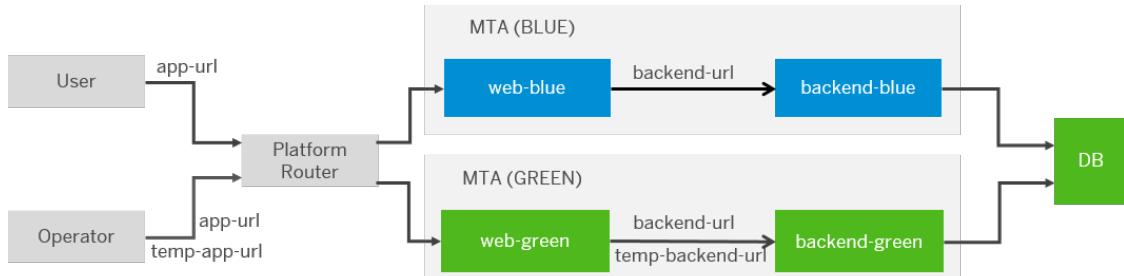
### ⓘ Note

This action does not perform a rollback and the state of apps, routes and services remains unchanged. Depending on your needs, you might want to remove the new app versions and the temporary routes.

4. To make the “green” version productive, choose [Resume](#).

This action:

- maps the productive routes to your green versions



- deletes the temporary routes
- restarts “green” apps with productive route configurations
- restarts the applications, so that the environment URLs are updated
- deletes the “blue” applications, which were productive before



## Results

For more detailed information and example results of the legacy blue-green deployment process, see [Legacy Blue-Green Deployment](#).

## Related Information

[Blue-Green Deployment of Multitarget Applications \[page 657\]](#)

[Blue-Green Deployment Strategy \[page 660\]](#)

[\(Experimental\) Incremental Blue-Green Deployment Strategy \[page 663\]](#)

### 4.1.8.1.9.2.2 Blue-Green Deployment Strategy

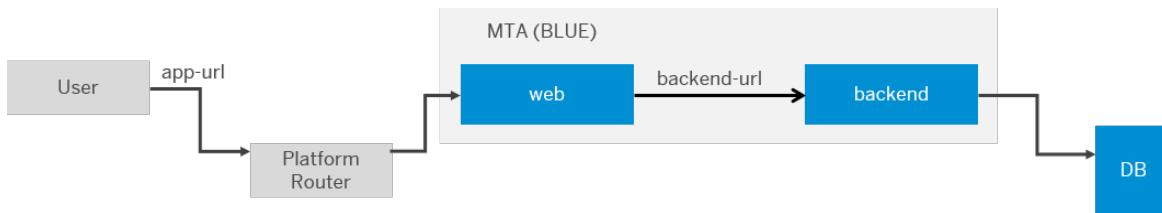
Use the current blue-green deployment of Multitarget applications.

## Prerequisites

### ⚠ Restriction

Blue-green deployment is supported only for Cloud Foundry applications. It is not supported for bound services, such as service instances and their configuration, workflow content, and HTML5 repository content, among others. Live and idle applications are bound to the same service instances.

- You have a previously deployed MTA, with functional productive applications and routes:



- You have ensured that the changes in the new version of the application are compatible with the old version. This includes changes in the database tables, UAA configurations, and so on.

## Context

If you have already executed successful deployments using the [Legacy Blue-Green Deployment \[page 658\]](#) method ('`cf bg-deploy`'), you should have no problems when executing deployments with the blue-green deployment strategy ('`cf deploy --strategy blue-green`'). However, this does not apply the other way around and it is generally not recommended to go back to the legacy blue-green deployment method.

## ⚠ Caution

Do not begin a blue-green deployment using '`cf bg-deploy`' and continue with '`cf deploy --strategy blue-green`', or vice versa, as this might result in downtime.

## Procedure

1. Deploy your updated MTA in idle state by executing the command `cf deploy <your-mta-archive-v2> --strategy blue-green`.

The first action is that all MTA services are updated.

This creates:

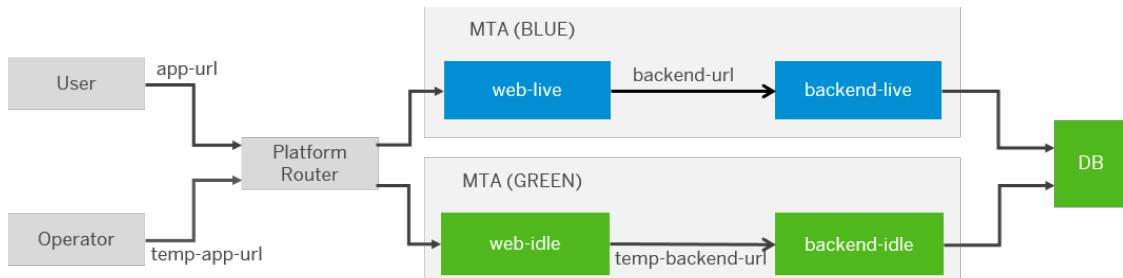
- new applications adding “idle” to the original application names

### ⓘ Note

The new application version is launched on all instances defined in the MTA deployment descriptor. During the testing phase or when the deployment is paused for user verification, the application temporarily requires double the usual resources (instances). This is because both the old and the new versions of the application are running concurrently. If the testing phase is skipped or if the deployment continues without delay, the resource usage will be optimized sooner.

If you are looking for a more resource-efficient deployment approach, see [\(Experimental\) Incremental Blue-Green Deployment Strategy \[page 663\]](#). However, keep in mind that this approach requires more time than the standard blue-green deployment strategy.

- temporary routes to the idle applications



- an interrupt to the process showing a message similar to the following:

### ⇢ Output Code

```
Process has entered testing phase. After testing your new deployment
you can resume or abort the process.
Use "cf deploy -i 7e6233b9-6001-11ea-8959-eeee0a98a87d -a abort" to
abort the process.
Use "cf deploy -i 7e6233b9-6001-11ea-8959-eeee0a98a87d -a resume" to
resume the process.
Hint: Use the '--skip-testing-phase' option of the deploy command to
skip this phase
```

2. Optionally, test the idle version of the application using the temporary routes.

### ⓘ Note

You can skip this step by using one of the following command line options:

- `--skip-testing-phase` - you have to use it when starting the process.
- `--skip-idle-start` - this option will also skip the start of the newly deployed applications on idle routes.

3. If you do not want to make the “idle” version available, abort the process using `cf deploy -i <operation ID> -a abort`.

### ⓘ Note

This action does not perform a rollback and the state of apps, routes and services remains unchanged. Depending on your needs, you might want to remove the new app versions and the temporary routes.

4. If you want to make the “idle” version productive, manually resume the process, using `cf deploy -i <operation ID> -a resume`.

This does the following:

- maps the productive routes to your idle versions
- deletes the temporary routes
- restarts “idle” apps with productive route configurations
- deletes the “live” applications, which were productive before



## Results

For more detailed information and example results of the blue-green deployment process, see [Blue-Green Deployment](#).

## Related Information

[Blue-Green Deployment of Multitarget Applications \[page 657\]](#)

[Legacy Blue-Green Deployment \[page 658\]](#)

[\(Experimental\) Incremental Blue-Green Deployment Strategy \[page 663\]](#)

[Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#)

### 4.1.8.1.9.2.3 (Experimental) Incremental Blue-Green Deployment Strategy

Use the incremental blue-green deployment strategy to save resources by incrementally deploying the new version of your Multitarget application.

#### General Information

##### ⚠ Restriction

Blue-green deployment is supported only for Cloud Foundry applications. It is not supported for bound services, such as service instances and their configuration, workflow content, and HTML5 repository content, among others. Live and idle applications are bound to the same service instances.

During the incremental blue-green deployment, one instance of the live application is stopped and one new instance of the new application is started at a time, until the new application is started on all instances. This strategy ensures a smooth transition between the old and new versions of an application. The process consists of the following steps:

1. **Initial Setup:** The new version of the application is deployed to only one instance.
2. **Testing Phase:** The deployment process enters the validation phase, where users or automated systems can test the new application.
3. **User Acceptance:** Customers or designated testers evaluate the new application.
4. **Deployment Continuation:** Once testing results are satisfactory, the deployment process resumes.
5. **Incremental Scaling:**
  - One instance of the old application is stopped.
  - One instance of the new application is started.This step is repeated until the new application reaches the desired number of instances.

##### ⓘ Note

During the incremental blue-green deployment, the total number of instances (old + new) will not exceed **N + 1**, where **N** is the desired instance count for the new application.

If an instance crashes during this step, a "rollback" procedure is initiated:

- The old application is scaled back up to its initial instance count.
- The new application is scaled down to one instance.

This rollback ensures system stability by restoring the previous state of the applications.

6. **Completion:** After the new application is fully scaled up, the remaining instances of the old application are stopped and removed.

##### ⓘ Note

If you are using the [Application Autoscaler](#) service, note the following:

- The new application will not be autoscaled during the incremental blue-green deployment process. The autoscaling will start after the deployment process completes.

- Once the deployment process resumes and enters the **Incremental Scaling** phase, autoscaling will also be disabled for the live application.

This ensures that neither the idle nor the live applications are rescaled by the Autoscaler, preventing potential inconsistencies.

The incremental blue-green deployment has the following advantages and disadvantages:

Pros	Cons
The incremental blue-green deployment requires less memory resource than the standard blue-green deployment, which means that it could save memory quota and reduce costs.	The incremental blue-green deployment requires more time to complete than the normal blue-green deployment, as the new application instances are started sequentially, which means that it waits for every new instance to completely start before starting the next one.
In certain scenarios, services may calculate monthly costs based on the maximum number of instances associated with an application. In a standard blue-green deployment, where both versions of the application run simultaneously, this can temporarily double the number of instances, potentially leading to higher charges.	
It is advisable to consult with the service provider, to understand the specific pricing implications for end customers.	

To perform an incremental blue-green deployment, follow the steps in the sections below.

## Prerequisites

- You have a previously deployed MTA, with functional productive applications and routes:



- You have ensured that the changes in the new version of the application are compatible with the old version. This includes changes in the database tables, UAA configurations, and so on.

## Context

If you have already executed successful deployments using the [Blue-Green Deployment Strategy \[page 660\]](#) (`cf deploy --strategy blue-green`), you should have no problems when executing deployments with the incremental blue-green deployment strategy (`cf deploy --strategy incremental-blue-green`). This also applies if you decide to go back to the blue-green deployment strategy.

However, this does not apply for the [Legacy Blue-Green Deployment \[page 658\]](#) (`cf bg-deploy`). It is not recommended to go back to this deployment method.

### ⚠ Caution

Do not begin a blue-green deployment using '`cf bg-deploy`' and continue with '`cf deploy --strategy blue-green / incremental-blue-green`', or vice versa, as this might result in downtime.

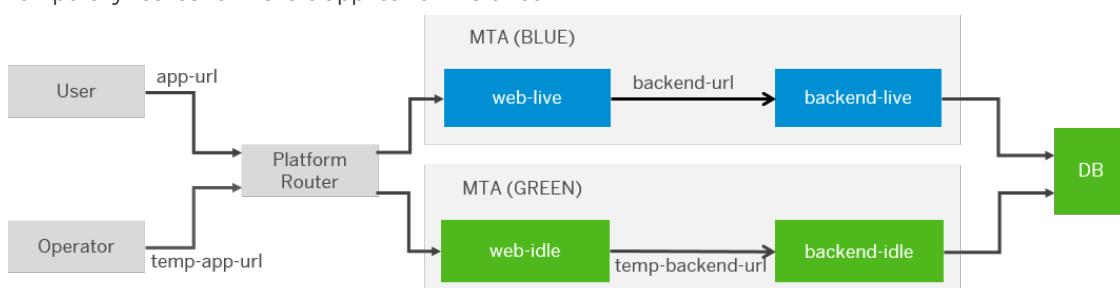
## Procedure

1. Deploy your updated MTA in idle state by executing the command `cf deploy <your-mta-archive-v2> --strategy incremental-blue-green`.

The first action is that all MTA services are updated.

This creates:

- one instance of the new application with the suffix "idle" added to the original application name
- temporary routes to this idle application instance



- an interrupt to the process showing a message similar to the following:

### ↔ Output Code

```
Process has entered testing phase. After testing your new deployment
you can resume or abort the process.
Use "cf deploy -i 7e6233b9-6001-11ea-8959-eeee0a98a87d -a abort" to
abort the process.
Use "cf deploy -i 7e6233b9-6001-11ea-8959-eeee0a98a87d -a resume" to
resume the process.
Hint: Use the '--skip-testing-phase' option of the deploy command to
skip this phase
```

2. Optionally, test the idle version of the application using the temporary routes.

### → Tip

You can skip this step by using one of the following command line options:

- `--skip-testing-phase` - this option will not wait for user confirmation after testing.
- `--skip-idle-start` - this option will also skip the start of the newly deployed application on idle routes and will not wait for user confirmation.

3. If you do not want to make the "idle" version available, abort the process by using `cf deploy -i <operation ID> -a abort`.

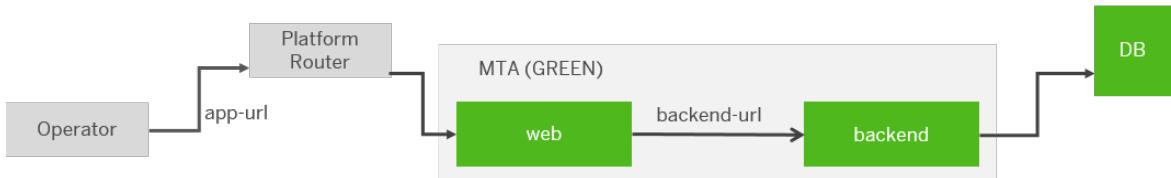
### ⓘ Note

This action does not perform a rollback and the state of apps, routes, and services remains unchanged. Depending on your needs, you might want to remove the new app versions and the temporary routes.

4. If you want to make the “idle” version productive, manually resume the process by using `cf deploy -i <operation ID> -a resume`.

This does the following:

- maps the productive routes to your idle application versions
- deletes the temporary routes of the idle application
- restarts idle apps with productive route configurations
- incrementally stops one instance of the live application and starts a new instance of the new application until it is started on all required instances
- deletes the live applications which were productive before



## Examples

For more detailed information and example results of the incremental blue-green deployment process, see [Blue-Green Deployment with incremental-blue-green strategy](#).

## Related Information

[Blue-Green Deployment Strategy \[page 660\]](#)

[Legacy Blue-Green Deployment \[page 658\]](#)

[Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#)

## 4.1.8.1.9.3 (Experimental) Namespaces

Use this feature to prevent conflicts for applications deriving from the same MTA, but with different version, features, and configuration.

### ⚠ Restriction

- This feature is experimental. We advise you first try it out in a non-productive environment.

- The namespace can only be passed as an argument to the deployment call, and cannot be modelled directly in the descriptor.
- The namespace cannot be longer than 36 characters.

## Prerequisites

- Make sure the version of your MultiApps Plug-in is 2.5.0 or higher. If required, use the procedure described in [Install the MultiApps CLI Plugin in the Cloud Foundry Environment \[page 2455\]](#) to update it.

## Feature Description

By using the namespaces feature, you can deploy and operate a single multitarget application multiple times within a single space. This is useful when you have to avoid conflicts among applications or entities. It modifies the modules and resources by adding a namespace at the beginning of the application names, service names, and routes.

The feature is employed by adding the parameter `--namespace`, when you input the deployment command in the Cloud Foundry Command Line Interface (CF CLI), and by modifying the deployment descriptor as described below. Using a more detailed configuration, you can include or exclude specific resources from having the namespace feature applied to them, namely:

- resources
- modules
- routes.

When set so, the excluded resources are shared among all namespaces of this MTA archive.

## Fine-Grained Configuration

You can configure the application of namespaces at different levels according to your needs. When the same `apply-namespace` parameter is configured in several places for one MTA deployment, a certain order of priority applies. This order provides flexibility if you want to set a namespace for all applications, application routes, and services that are part of the MTA, but at the same time, you need to define a specific namespace for a particular application, application route, or service.

The priority for `apply-namespace` parameters in descending order is the following:

1. **Operation parameters / Command-Line Options (Highest Priority)**

Operation parameters passed during the MTA deployment have the highest priority. If you provide an `apply-namespace` parameter as a command-line option, it overrides any other parameters defined in the MTA descriptor. Operation parameters apply to all applications.

There are four available options for applying a namespace:

- `--apply-namespace-app-names true/false`: Applies namespace to application names.

- `--apply-namespace-service-names true/false`: Applies namespace to service names.
- `--apply-namespace-app-routes true/false`: Applies namespace to application routes.
- `--apply-namespace-as-suffix true/false`: Applies namespace either as prefix or suffix.

**Example:**

When you deploy using the command `cf deploy ... --namespace test --apply-namespace-app-names true --apply-namespace-service-names false --apply-namespace-app-routes true`, all application names and application routes will have the prefix 'test-'. Service names will remain unchanged, as specified by the `--apply-namespace-service-names false` option. Other `apply-namespace` parameters will not affect this configuration.

If you use the command `cf deploy ... --apply-namespace-app-names true --apply-namespace-service-names false --apply-namespace-app-routes true` without the '`--namespace`' option, the `apply-namespace` options will be ignored.

For more information about `apply-namespace` options, refer to the **Command Options Overview** table in [deploy](#).

## 2. Module-Level and Resource-Level Parameters

If no command-line argument is provided, the module-level and resource-level parameters are considered next.

**Example:**

 Sample Code

```
modules:
  - name: java
    .....
    parameters:
      apply-namespace: true
      routes:
        - route: host.some-domain.com
          apply-namespace: true
resources:
  - name: my-service
    .....
    parameters:
      apply-namespace: false
```

## 3. Global-Level Parameters

If neither command-line options nor module-level parameter are provided, the global parameters are used. Global parameters are applicable for all applications.

 Sample Code

```
parameters:
  apply-namespace:
    app-names: true
    service-names: false
    app-route: true
```

## 4. Default Values

If you don't provide any of the above mentioned parameters, the application will use the predefined default values. When you set the '`--namespace`' option, the default value is `true`, otherwise, it's `false`.

## Examples

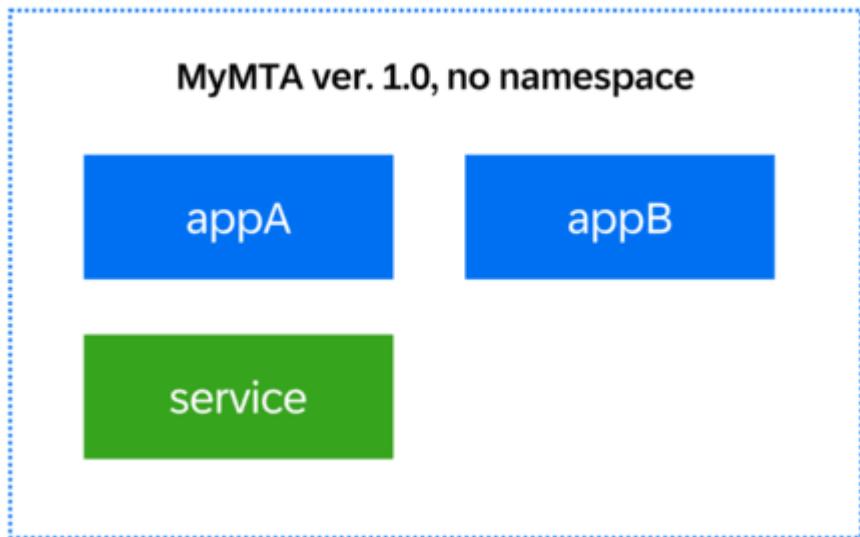
To deploy several entities of a multitarget application using namespaces, follow these steps:

1. Using the CF CLI, deploy the initial MTA archive by using the command `cf deploy ./MyMTA.mtar` with the descriptor given below:

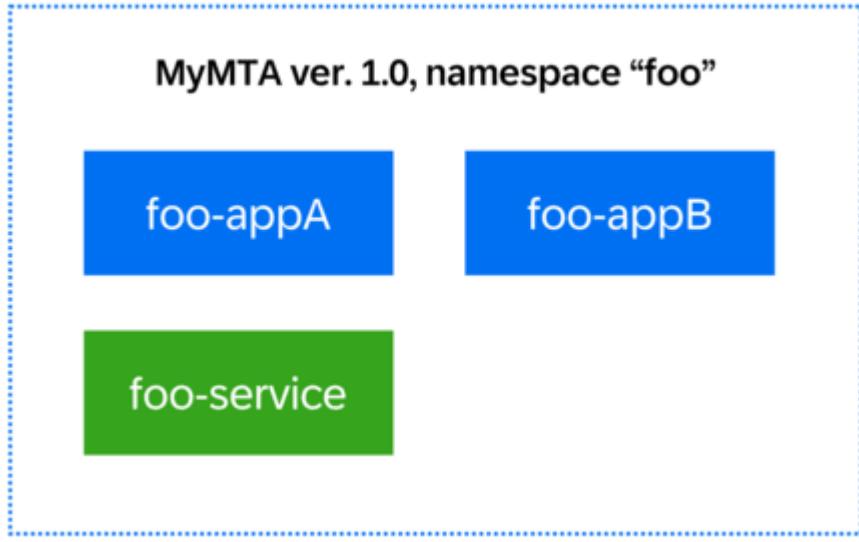
### Sample Code

```
modules:  
  - name: appA  
    type: java.tomee  
    path: application.war  
    requires:  
      - name: service  
    parameters:  
      routes:  
        - route: appA.${default-domain}  
        - route: route-without-namespace.${default-domain}  
  - name: appB  
    type: java.tomee  
    path: application.war  
resources:  
  - name: service
```

The result should match the image below:



2. Deploy again the MTA archive by using the command `cf deploy ./MyMTA.mtar --namespace foo` with the same descriptor. The result should match the image shown below:



If you bind 'appA' to 'appA.my-domain.com', the system automatically binds 'foo-appA' to 'foo-appA.my-domain.com' upon deployment. If 'appA' requires a service instance called 'service', 'foo-appA' binds to a different service instance named 'foo-service'.

There are also other ways to achieve this result, for example:

- Deploy using the `cf deploy` command with the following options:

```
cf deploy ./MyMTA.mtar --namespace foo --apply-namespace-app-names true --apply-namespace-service-names true --apply-namespace-app-routes true
```

This command deploys the MTA archive 'MyMTA.mtar' and applies the namespace 'foo' to the application names, service names, and application routes.

- Set global parameters in the deployment descriptor:

If you don't have operation parameters or module-level and resource-level parameters, you can set the global parameters in the deployment descriptor to `true`. This approach produces the same result as using the `cf deploy` command with the namespace options.

Both methods ensure that the namespace is applied consistently across application names, service names, and application routes during the deployment process.

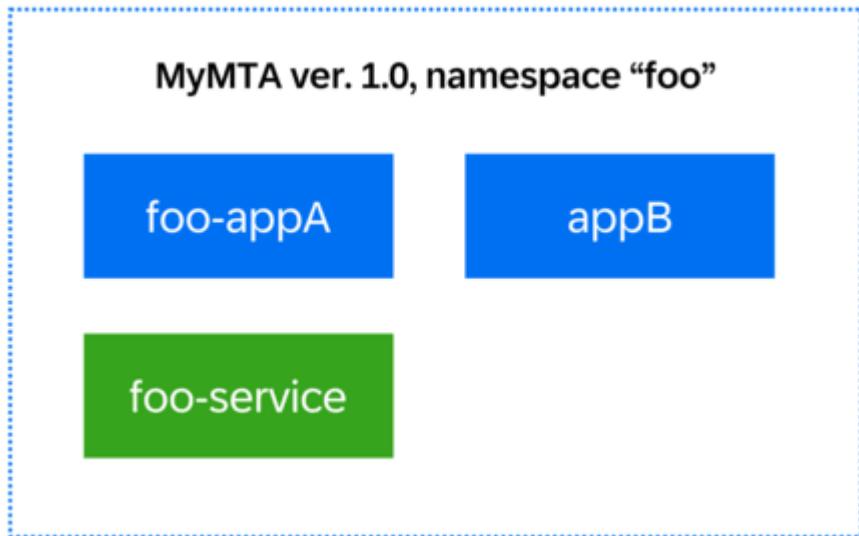
3. Deploy again the MTA archive by using the command `cf deploy ./MyMTA.mtar --namespace foo` with the given descriptor below:

#### ↔ Sample Code

```
modules:
  - name: appA
    type: java.tomee
    path: application.war
    requires:
      - name: service
    parameters:
      routes:
        - route: appA.${default-domain}
          apply-namespace: true
  - name: appB
    type: java.tomee
    path: application.war
    parameters:
      apply-namespace: false
resources:
  - name: service
```

```
parameters:  
  apply-namespace: true
```

The result matches the image below:



The deployment automatically binds 'foo-appA' to 'foo-appA.my-domain.com'.  
If the descriptor contains:

```
routes:  
  - route: appA.${default-domain}  
    apply-namespace: false
```

then 'foo-appA' automatically binds to 'appA.my-domain.com' upon deployment instead.

## Special Use: Sharing One Resource Among Several MTAs Using Namespaces

You can use deployment with namespaces in a way that only one of the resources receives a namespace. To do so, you use the parameter `apply-namespace` in the MTA deployment descriptor follows, and then deploy by using the procedure above. The deployment result should be as follows:

For the example below we use a service, but this special use can be applied to any module or resource in the descriptor.

### ↔ Sample Code

```
...  
  - name: serviceA  
    type: auditlog  
    parameters:  
      apply-namespace: false  
...
```

In the snippet above, the `apply-namespace: false` overrides the default value `true`, thus prohibiting the creation of additional instances of the service. This means that the service is unique in the space across all deployments of the MTA, and shared among them.

To see a practical example of the implementation above, go to [Deploying MTA in a Namespace](#).

#### 4.1.8.1.9.4 Parallel Module Deployment

In some cases, it is crucial that modules and therefore applications are deployed in a predictable and consistent order. To ensure this, the module-level attribute `deployed-after` has been introduced. It contains a list of module names. If a module has this attribute, it will be deployed only after the modules specified in the attribute have already been deployed.

The relations described through this attribute are transitive, so if module A should be deployed after module B, and module B should be deployed after module C, then it means that module A should be deployed after module C.

##### ↔ Sample Code

MTA Deployment Descriptor (`mtad.yaml`)

```
ID: com.sap.sample
version: 0.1.0
_schema-version: "3.2.0"
parameters:
  enable-parallel-deployments: true

modules:
  - name: ui
    type: javascript
    deployed-after: [ backend, metrics ]

  - name: backend
    type: java
    deployed-after: [ hdi-content ]
    requires:
      - name: metrics
        properties:
          METRICS_URL: ~{url}

  - name: metrics
    type: javascript
    deployed-after: [ hdi-content ]
    provides:
      - name: metrics
        properties:
          METRICS_URL: ~{url}

  - name: hdi-content
    type: hdi
```

In the example above, the `deployed-after` attributes guarantee that the `ui` module is deployed after the `backend` and the `metrics` modules, and they in turn are deployed after the `hdi-content` module. Note that the deployment order of the `backend` and the `metrics` modules is not specified in the attributes. This means that they can be deployed in any order.

## Parallel Deployment

### ⓘ Note

The `deployed-after` parameter is supported from major schema version 3 onwards.

In the example above, we have also specified the global MTA parameter `enable-parallel-deployments` with a value `true`. It activates the parallel deployment of MTA modules that do not have any deployment order dependencies between each other. If the parameter is missing or its value is `false`, the module deployment will be sequential.

### ⓘ Note

The `enable-parallel-deployments` parameter has influence only on modules deployment, see section [Sequential Resource Processing](#) for the resources process.

### ⓘ Note

The `enable-parallel-deployments` parameter is supported from major schema version 3 onwards.

## Circular Dependencies

Due to a modelling error, the user can introduce direct or transitive circular deployment order dependencies between modules. In such cases, this will be reported as a deployment error.

## Compatibility with Previous Deployment Order

The previous deployment order algorithm was based on the `requires` module-level attribute, which contains a list of module names or provided dependencies from other modules. Since it is also the way to model configuration dependencies, this mechanism was not explicit enough to model a deployment order .

There are many applications that are still depending on the old deployment order algorithm. To support them until they adapt to the new modeling, the new deployment order is introduced in a backward compatible manner. This means that if there are no `deployed-after` module-level elements in the MTA descriptor and the global MTA parameter `enable-parallel-deployments` is set to `false` or is missing, the old ordering algorithm is enabled by default.

## 4.1.8.1.9.5 Sequential Resource Processing

In various cases, some resources depend on other resources to already exist in order to process. Whenever an order of resource processing is necessary, the `processed-after` attribute can be utilized.

This attribute contains a list of resource names. It creates a sequence of services, an order, in which the services should be processed. If a resource has this attribute, it will be processed after the other resources in a higher position are processed.

### ⓘ Note

If the parameter `processed-after` is not specified, the resources are processed in parallel.

### ⓘ Note

The `processed-after` parameter is supported from schema version 3 onwards. If you attempt to use `processed-after` with schema version lower than 3, the deployment will ignore the attribute.

**Example:** Service A requires service B and service C to be processed first. On the other hand, both service B and service C require service D to be processed first. Service D does not require any other service to be processed. So the sequence for processing is - D → B, C → A.

### ↴ Sample Code

```
MTA Deployment Descriptor (mtad.yaml)

_schema-version: 3
ID: multiple-anatz
version: 3.0.0

modules:
- name: multiple-anatz
  type: staticfile
resources:
- name: serviceA
  type: org.cloudfoundry.managed-service
  parameters:
    service: application-logs
    service-plan: lite
  processed-after: [ serviceB, serviceC ]

- name: serviceB
  type: org.cloudfoundry.managed-service
  parameters:
    service: application-logs
    service-plan: lite
  processed-after: [ serviceD ]

- name: serviceC
  type: org.cloudfoundry.managed-service
  parameters:
    service: application-logs
    service-plan: lite
  processed-after: [ serviceD ]

- name: serviceD
  type: org.cloudfoundry.managed-service
  parameters:
    service: application-logs
    service-plan: lite
```

In the example, the `processed-after` attribute ensures that serviceD is processed first, then serviceC and serviceB are processed in parallel, and finally serviceA is processed.

## Circular Dependencies

Due to a modelling error, you can create direct or transitive circular deployment order dependencies between resources. In such cases, it is reported as a deployment error.

### 4.1.8.1.9.6 Features Related to SAP Alert Notificaton service for SAP BTP

Deploy and update the SAP Alert Notification service for SAP BTP using the multitarget application concept, or get notified by the service about multitarget application operation status.

#### Deploying and Updating the service

As an alternative to the manual procedure, you can use a deployment descriptor to automate the initial deployment of SAP Alert Notification service. You can use the same procedure to update the service parameters. See the procedure described in [Configuration Management Using Multitarget Application Descriptors](#).

#### Using the SAP Alert Notification service for BTP for alerts during deployment and undeployment of multitarget applications

You can set SAP Alert Notification service for SAP BTP to dispatch alerts to a communication channel of your choice, so that you are informed about the status of MTA deployment and undeployment operations. For example, if setup to do so, upon a successful deployment it can send you an e-mail or a Slack message. You can also get notified for problems in MTA operations, as you can also receive alerts if the deployment has encountered issues.

The integration between the services is out-of-the-box, that is, no special steps are required for its activation. If you have a service instance in the space where the MTA deployment takes place, it would automatically receive alerts from the MTA deployments in that space.

See how you can match events using multitarget operations as described in [Multitarget Application Events](#).

To see additional details, see [Receive instant notifications...with the SAP Cloud Deployment service!\[\]\(126aceaa099d6c1c1542df5a081babba\_img.jpg\)](#).

## 4.1.8.1.9.7 Content Deployment

SAP Business Technology Platform Cloud Foundry supports the lifecycle of all standard Cloud Foundry entities, such as applications and services. To satisfy all business requirements, SAP Business Technology Platform Cloud Foundry also supports the lifecycle of other entities that do not match 1-to-1 to CF native entities.

SAP traditionally provides its customers with complex business solutions, which work with their own entities and content. Every service might have its own content with a specific format that is developed, provided, or consumed by end customers.

SAP Workflow Management provides the possibility for SAP partners to develop their own BPM workflows (for example, Leave Request flow) and provide them to end customers. In turn, end customers can easily import and use the developed BPM workflows.

With multitarget applications, you can manage the lifecycle of content provided by different SAP applications. This way, by using one standard approach via an MTA, you can deploy various kinds of SAP application content in your accounts, normally done on the CF space level.

### ⚠ Restriction

Content deployment used with [blue-green deployment](#) will be done during the first phase when idle suffixes and temporary routes are used. The same rule is applicable for all content deployment mechanisms – either with direct content deployment or with content deployers (`hdi-deploy`, `html5-app-deployer`, `portal_deployer`). HTTP Destinations, created with direct content deployment, will not be updated after the validation phase, so avoid using default resolvable parameters  `${default-host}`,  `${default-uri}`,  `${default-url}`. Configurations that include idle suffixes and temporary routes may persist after the MTA deployment finishes. Content that was already deployed during the first phase cannot be reverted or removed. It will be necessary to trigger a new MTA deployment.

## Supported Content Deployment Mechanisms

- Using content applications – there is a dedicated SAP provisioned content deployer for each content type that communicates with the dedicated content backend
  - [DEPRECATED] Deploying Content with Simulated App Execution
  - Deploying Content with CF task execution
- Using direct content deployment – there is a direct communication between the SAP Cloud Deployment service and the content backends
  - Deploying Content with Generic Application Content Deployment
  - CPI content deployment

## Supported MTA Module Types for Content Deployment

Module Type	Content Deployment Mechanism
com.sap.portal.site-content	Simulated App Execution
com.sap.html5.application-content	Simulated App Execution
com.sap.portal.content	CF task execution
com.sap.html5.application.content	CF task execution
com.sap.xs.hdi	CF task execution
com.sap.application.content	Generic Application Content Deployment
com.sap.integration	CPI content deployment
com.sap.api.management	CPI content deployment

For all predefined parameters and properties for each MTA module type, see [MTA Module Types \[page 595\]](#).

### 4.1.8.1.9.7.1 Deploying Content with Generic Application Content Deployment

This approach provides a mechanism for direct content deployment from SAP Cloud Deployment service to the content backend without the need for an intermediate Cloud Foundry application.

#### ⓘ Note

This approach is supported only for MTA schema version 3.1 and higher.

#### ⓘ Note

This approach is recommended for its light-weight nature, minimal footprint in your Cloud Foundry account and avoided overhead of transient content applications.

#### ⓘ Note

The examples below are partial and they intend to show the different capabilities of content deployment. For specific application content, see the dedicated documentation linked below.

The mechanism utilizes the Generic Application Content Deployment protocol (GACD). It allows you to deploy content represented by an MTA module of type com.sap.application.content to a certain list of supported services.

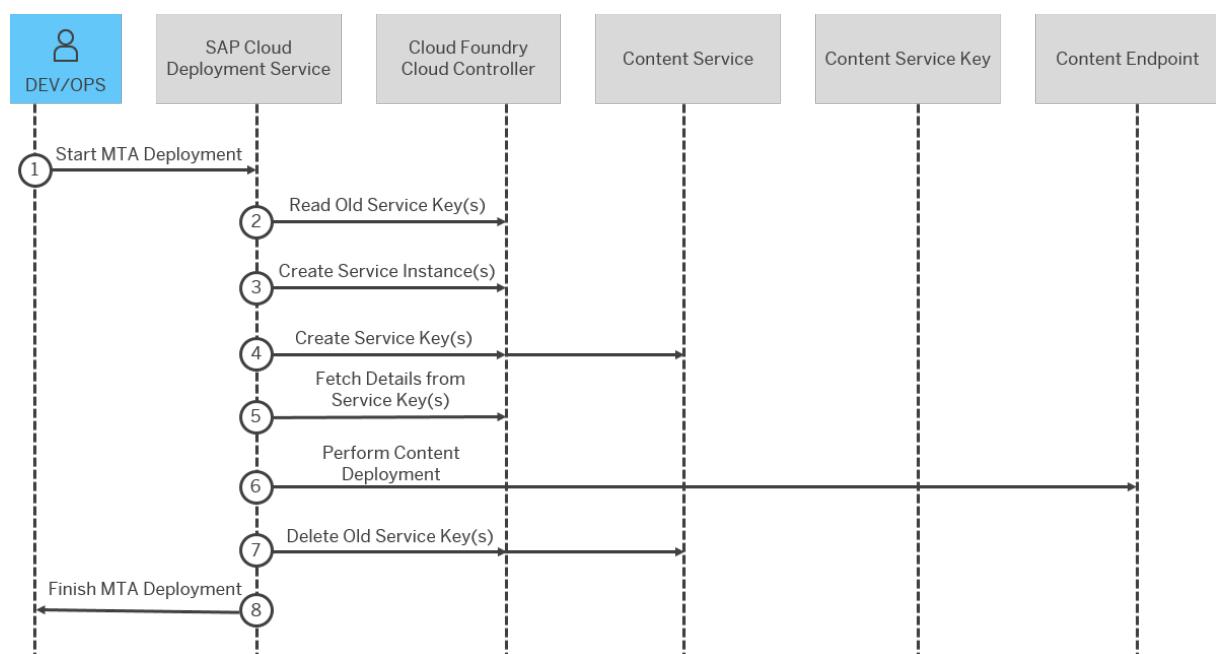
## Supported service offerings

Cloud Foundry service offerings that currently support GACD are:

- `html5-apps-repo` (For more information, see [Deploy Content Using Generic Application Content Deployer](#))
- `portal` (For more information, see [Add an SAP Fiori Launchpad Module](#))
- `workflow` (For more information, see [Create a Workflow Module](#))
- `destination` (For more information, see [Create Destinations Using the MTA Descriptor](#))

## How Content is Deployed with Generic Application Content Deployment

The diagram below illustrates the steps and BTP components involved in the process of content deployment with Generic Application Content Deployment:



The following steps describe the process of content deployment with Generic Application Content Deployment:

1. A developer/operator runs MTA deployment – the flow demonstrates only modules of type `com.sap.application.content` indicating content (for example - a single MTA module and multiple required services, only one of them is marked with parameter `content-target: true`).
2. SAP Cloud Deployment service prepares for the next steps. This step includes:
  1. SAP Cloud Deployment service gets the existing services and service keys which are related to previous MTA deployment. The list comprises of a content service instance, a content service key and eventually additional service keys.
  2. Based on the name, SAP Cloud Deployment service decides which of the existing service keys will be deleted, which will be kept, and whether new ones will be created. In the end, only one of them will be used as a content target.
3. SAP Cloud Deployment service creates or reuses a service instance or instances that are relevant for the content deployment, not only the one that serves as a content target.
4. SAP Cloud Deployment service creates or reuses a service key. This step might include the creation of several service keys needed for successful content deployment, not only the one that serves as a content target.

5. SAP Cloud Deployment service fetches the credentials of the required service key(s). In this step content endpoint is detected from content target service key. The credentials of all remaining service keys serve as configurations for the next step.
6. SAP Cloud Deployment service executes the content deployment to content endpoint and passes previously collected data as content configurations.
7. SAP Cloud Deployment service deletes the discontinued service keys calculated in step 2. Normally these are service keys that are not part of the current MTA and were used in the previous deployment.
8. MTA deployment completes and SAP Cloud Deployment service reports the status to the developer/operator.

## Content Target

When an MTA module of type `com.sap.application.content` is present in your MTA, SAP Cloud Deployment service processes it and delivers its module content to the content target.

The content target is explicitly set with a flag in the MTA and it is unknown to the SAP Cloud Deployment service without it.

There should be only one content target defined per MTA module of type `com.sap.application.content`. This guarantees that the content will be deployed to only one place.

The content target is defined when a content module requires a service or service key, and in that required dependency the parameter `content-target` is set to `true`. In case of a missing or duplicated `content-target` parameter in more than one required dependency, the deployment will not proceed.

Normally, content modules require MTA resources that represent Cloud Foundry service instances or service keys. When multiple service instances or keys are required, they are used by default for content deployment configuration. For more information see the [Configuration from other service instances/keys \[page 683\]](#) section.

## MTA Modelling

### Deliver file content

You can use a file or directory that is treated as content. The file or directory is part of the MTA archive, and its path is defined in the `MANIFEST.MF` file.

The following code block contains a snippet from an MTA deployment descriptor (`mtad.yaml`):

#### ↔ Sample Code

```
_schema-version: '3.1'
ID: com.sap.example.content.deployment
version: 0.0.1
modules:
  - name: content-module
    type: com.sap.application.content
    requires:
      - name: workflow_service
        parameters:
          content-target: true
```

```

resources:
- name: workflow_service
  type: org.cloudfoundry.managed-service
  parameters:
    service-plan: standard
    service: workflow

```

The following code block contains the accompanying `MANIFEST.MF` file:

#### ↔ Sample Code

```

Name: content-module/data
MTA-Module: content-module
Content-Type: application/zip

```

### **Delivery text / Inline content**

You can enter content definitions inline in the MTA descriptor as part of module parameter `content`. In the example below the template for new destinations (`foo-api` and `bar-api`) will be deployed to `destination-service`:

#### ↔ Sample Code

```

modules:
- name: destination-content
  type: com.sap.application.content
  requires:
  - name: destination-service
    parameters:
      content-target: true
  - name: foo-api
  - name: bar-api
  parameters:
    content:
      subaccount:
        destinations:
        - Name: foo-api
          URL: ~{foo-api/url}
          forwardAuthToken: true
        - Name: bar-api
          URL: ~{bar-api/url}
          forwardAuthToken: true
    <....>
  resources:
- name: destination-service
  type: org.cloudfoundry.managed-service
  parameters:
    service: destination
    service-plan: lite

```

## **Service Keys in Content Deployment**

When an MTA module of type `com.sap.application.content` requires an MTA resource, the SAP Cloud Deployment service will create or re-use a service key.

Service keys in content deployment are an alternative of service bindings between applications and services. When there is an intermediate content application, it might be bound to different services that provide specific

capabilities. In direct content deployment, the required service keys have the same role. For example, a content module might require an xsuaa service instance, which might be used for secure communication at a later stage.

By default, service keys to all required services are created with the name that follows the template <module\_name>-<resource\_name>-credentials. For example, the result of the sample below will be a service instance with name workflow\_service which will have a service key named content-module-workflow\_service-credentials:

#### ↔ Sample Code

```
modules:
- name: content-module
  type: com.sap.application.content
  requires:
    - name: workflow_service
      parameters:
        content-target: true
  resources:
    - name: workflow_service
      type: org.cloudfoundry.managed-service
```

By default, the existing service keys are re-used in future deployments. This is done to save time from their re-creation and avoid potential issues if they are used outside of the MTA and their credentials are invalidated.

### Customizing service keys

You can customize the service keys created as part of the content deployment by using the required dependency parameter service-key. This way, you can define the service key name or pass service key creation parameters:

#### ↔ Sample Code

```
modules:
- name: content-module
  type: com.sap.application.content
  requires:
    - name: workflow_service
      parameters:
        content-target: true
      service-key:
        name: workflow_service-key
        config:
          <map entries for creation parameters>
```

### Re-create custom service keys for deployment

This option extends the customization of service keys explained above. Based on its name, an existing service key may be re-used or deleted, as described in [How Content is Deployed with Generic Application Content Deployment \[page 678\]](#).

The option is particularly useful if you want to rotate and renew the credentials used for content deployment, like when they are based on user/password credentials or certificates.

You can set the service keys to automatically rotate for each new deployment by using the existing deployment parameter \${timestamp} as part of the name of the defined custom service key. Doing so ensures that the key will have a new name for each deployment, resulting in Deploy Service creating and using a new key for the related content deployment. The previously used service key will be deleted after the new one is in place.

## ↔ Sample Code

```
modules:
- name: content-module
  type: com.sap.application.content
  requires:
    - name: workflow_service
      parameters:
        content-target: true
        service-key:
          name: workflow_service-key-${timestamp}
```

## Using existing service keys

You can deploy content directly to an existing service key with a descriptor modeled appropriately. This option might be useful if you want to manage the lifecycle of the service key outside of the MTA:

## ↔ Sample Code

```
modules:
- name: content-module
  type: com.sap.application.content
  requires:
    - name: workflow_service_key
      parameters:
        content-target: true
resources:
- name: workflow_service_key #this service key needs to exist beforehand
  type: org.cloudfoundry.existing-service-key
  parameters:
    service-name: workflow_service #this service must be created and exist beforehand
```

## Using user-provided services

Sometimes, the content endpoint might not be exposed as a standard service broker in the BTP. In these cases, neither service instance nor service key can be created. The only option to utilize content deployment to such content target is via user-provided services.

## ↔ Sample Code

```
modules:
- name: content-module
  type: com.sap.application.content
  requires:
    - name: content-endpoint
      parameters:
        content-target: true
resources:
- name: content-endpoint
  type: org.cloudfoundry.user-provided-service
  parameters:
    content_endpoint: https://my-content-endpoint.com
```

## Content Deployment Configuration

Sometimes content deployment needs a configuration, which might be environment or context specific and cannot part directly in the content itself.

There are two supported ways to pass additional configuration for content deployment:

- Content Deployment Configuration from other service instances/keys
- Inline Content Deployment Configuration

### Configuration from other service instances/keys

By default, all required MTA resources from content modules are processed and used for content deployment configuration. This includes the MTA resources of types `org.cloudfoundry.managed-service`, `org.cloudfoundry.existing-service`, `org.cloudfoundry.existing-service-key` and `org.cloudfoundry.user-provided-service`. A service key is used for each of those resources, except for `org.cloudfoundry.user-provided-service`.

In the example below, the content-module requires two services and only one of them is marked as content-target. The other required service is xsuaa, which will be used for content configuration, and it will guarantee more secure communication during end-to-end flow:

#### ↔ Sample Code

```
_schema-version: '3.1'
ID: com.sap.example.content.deployment
version: 0.0.1
modules:
  - name: content-module
    type: com.sap.application.content
    requires:
      - name: xsuaa_service
      - name: html5_service
        parameters:
          content-target: true
resources:
  - name: html5_service
    type: org.cloudfoundry.managed-service
    parameters:
      service-plan: app-host
      service: html5-apps-repo
  - name: xsuaa_service
    type: com.sap.xs.aaa
```

When multiple service instances or keys are required, they are all used for content deployment configuration by default.

### Inline Content Deployment Configuration

You can provide content deployment configuration as part of the module parameter `config`:

#### ↔ Sample Code

```
modules:
  - name: content-module
    type: com.sap.application.content
    requires:
      - name: workflow_service
    parameters:
```

```

content-target: true
parameters:
  config: # the content deployment configuration
  env: dev
resources:
  - name: workflow_service
    type: org.cloudfoundry.managed-service
    parameters:
      service-plan: standard
      service: workflow

```

## Related Information

[Deploy Content Using the Generic Application Content Deployer \[page 394\]](#)

[Create Destinations Using the MTA Descriptor](#)

### 4.1.8.1.9.7.2 Deploying Content with CF Task Execution

This approach for content deployment relies on an intermediate Cloud Foundry application that communicates with the content backend. This application includes both the content and the content specific deployer.

The content specific deployer is defined as an application dependency during design time. It is then included in the MTA archive. During deployment, it serves as a client for the actual content deployment to the content backend.

#### Note

This approach is a natural successor to [Deploying Content with Simulated App Execution \[page 687\]](#).

When possible, it is recommended to use the current approach instead of the legacy one. Normally, there is only one dedicated content deployer for each content type that handles both approaches.

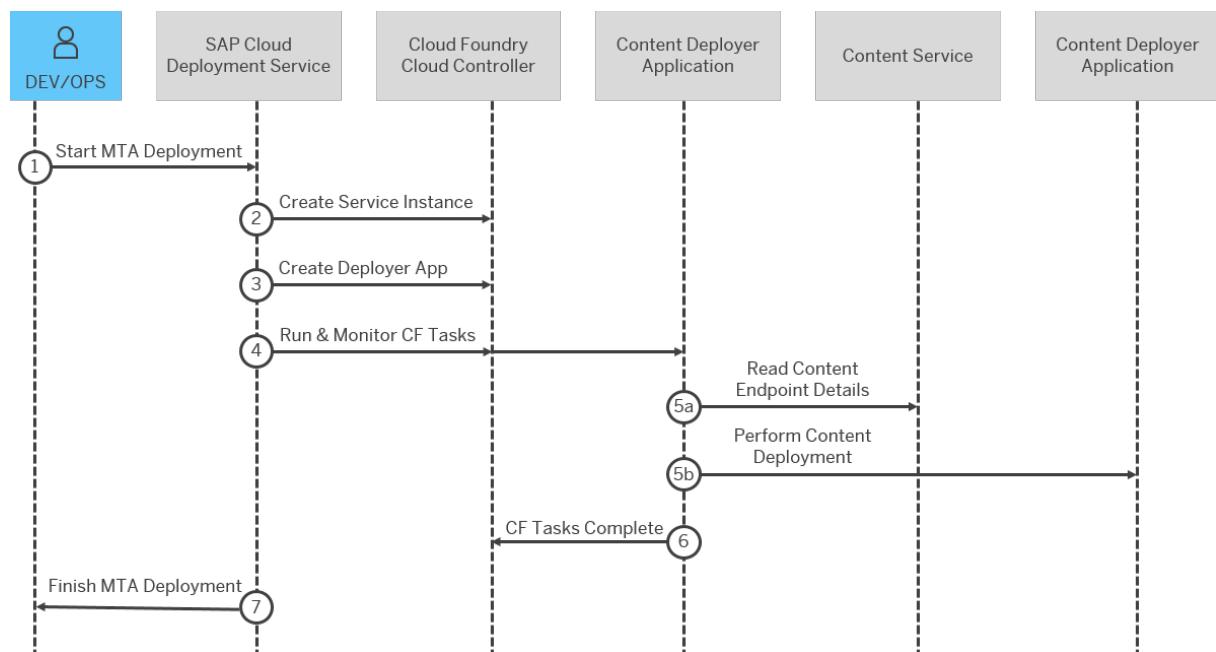
## Supported Content Types

Supported content types that currently allow deployment with CF Task execution:

- HDI content - MTA module type `com.sap.xs.hdi`. For more information, see [The SAP HDI Deployer](#)
- FLP content - MTA module type `com.sap.portal.content`
- HTML5 Content - MTA module type `com.sap.html5.application.content`. For more information, see [Deploy Content Using HTML5 Application Deployer](#)

## How Content is Deployed with CF Task Execution

The following diagram illustrates the steps and BTP components involved in the process of content deployment with CF task execution:



The following steps describe the process of content deployment with CF Task execution:

1. A developer/operator runs MTA deployment
2. SAP Cloud Deployment service creates or reuses a service instance provided by an SAP application
3. During this step:
  1. SAP Cloud Deployment service creates a Cloud Foundry application that serves as a content application
  2. SAP Cloud Deployment service binds the service instance to the content application
  3. SAP Cloud Deployment service uploads the combination of content and content deployer as application bits
4. SAP Cloud Deployment service runs CF tasks, including the one responsible for actual content deployment
5. The dedicated CF task of the content application triggers the content deployer, which in turn performs the actual content deployment. During this step:
  1. The CF task reads content details from the bound service instance and identifies the content endpoint and the required authentication
  2. The CF task executes the content deployment to the content endpoint
6. The CF task completed successfully
7. The MTA deployment completes and SAP Cloud Deployment service reports the status to the developer/operator

## Content Deployer and Service Instance

The Content Deployer and Service Instance follow the same rules as the ones from [Deploying Content with Simulated App Execution \[page 687\]](#).

## CF Tasks

Tasks are native to Cloud Foundry and are valuable in various scenarios, which demand one-off jobs. For more information, see [Tasks](#).

CF tasks, dedicated for content deployment, are predefined from the SAP Cloud Deployment service and must not be changed by the end customer. They are represented by the module level parameter `tasks`.

## MTA Modeling

Below you can find the modeling for supporting the content deployment scenario for MTA module type `com.sap.xs.hdi`:

### ↔ Sample Code

```
_schema-version: "3.1"
ID: my.app
version: "1.1"
- name: module-db
  type: com.sap.xs.hdi
  parameters:
    buildpack: nodejs_buildpack //Predefined parameter
    no-route: true // Predefined parameter - skipping route mapping to the
app
    no-start: true // Predefined parameter - skipping start of the app
because only tasks are important in the scenario
    memory: 256M // Predefined parameter
    tasks: // Predefined parameter - all details below define the CF tasks
      - name: deploy
        memory: 256M
        command: npm start // Shell script that will be used for triggering
actual content deployment
    properties:
      EXIT: 1 // Predefined property - instructs the HDI deployer to exit
when done
    requires:
      - name: hdi-service

resources:
  - name: hdi-service // Provides content endpoint for the actual content
deployment. Content endpoint could be different based on the used database
  type: com.sap.xs.hdi-container
```

### 4.1.8.1.9.7.3 Deploying Content with Simulated App Execution

This approach for content deployment relies on an intermediate Cloud Foundry application that communicates with the content backend. This application includes both the content and the content specific deployer.

The content specific deployer is defined as an application dependency during design time. It is then included in the MTA archive. During deployment it serves as a client for the actual content deployment to the content backend.

#### ⓘ Note

This approach is a legacy one. Over time it has proven to have some major drawbacks. If possible, migrate to [Deploying Content with Generic Application Content Deployment \[page 677\]](#) or at least to [Deploying Content with CF Task Execution \[page 684\]](#) of the corresponding module type.

#### ⓘ Note

All supported MTA module types for this approach will either be discontinued or will be transformed to use [CF Task Execution \[page 684\]](#).

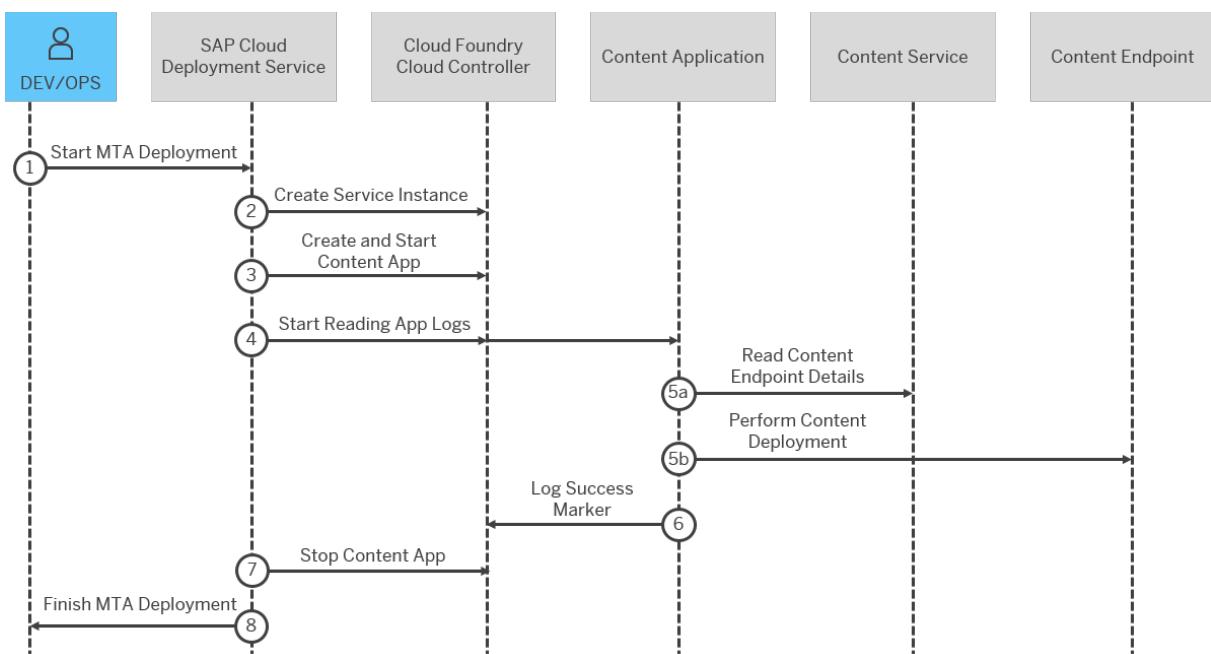
### Supported Content Types

Supported content types that currently allow deployment with simulated app execution:

- FLP content - MTA module type `com.sap.portal.site-content`
- HTML5 Application Content – MTA module type `com.sap.html5.application-content`. For more information see [Deploy Content Using HTML5 Application Deployer](#).

### How Content is Deployed with Simulated App Execution

The following diagram illustrates the steps and BTP components involved in the process of content deployment with simulated app execution:



The steps below describe the process of content deployment with simulated app execution:

1. A developer/operator runs MTA deployment.
2. SAP Cloud Deployment service creates or reuses a service instance provided by an SAP application.
3. The third step includes:
  1. SAP Cloud Deployment service creates a Cloud Foundry application that serves as a content application.
  2. SAP Cloud Deployment service binds the service instance to the content application.
  3. SAP Cloud Deployment service uploads the combination of content and content deployer as application bits.
  4. SAP Cloud Deployment service starts the application, which in turn starts the content deployer.
4. SAP Cloud Deployment service starts to read the application logs and searches for the success-marker and/or failure-marker values.
5. Content deployer performs the actual content deployment. This step includes:
  1. Content deployer reads content details from the bound service instance and identifies the content endpoint and required authentication.
  2. Content deployer executes the content deployment to content endpoint.
6. Via certain success-marker or failure-marker, content deployer reports the status as application logs.
7. SAP Cloud Deployment service stops the content application.
8. MTA deployment completes and SAP Cloud Deployment service reports the status to the developer/operator.

## Content Deployer and Service Instance

The content deployer and service instance are provided by the SAP business application that uses the business content. This way, the content deployer will know how to communicate to the backend on which the content will be deployed. Thus the SAP Cloud Deployment service is isolated from the actual content deployment protocol and implementation.

Normally, the content deployers are simple clients that implement specific REST APIs to pass the content. When the content application is started it invokes a content deployer which calls certain endpoints and executes the actual content deployment.

The content deployer identifies the content endpoint based on the service binding. Content endpoint could be any URL either provided by an application in BTP Cloud Foundry or one outside of the BTP entirely.

It is possible that a content application will have bindings to multiple service instances to complete the content deployment. For example, there could be a reference to an XSUAA service instance, which will ensure secure communication between the content deployer and content endpoint.

## Success Marker and Failure Marker

Success and failure markers are part of the protocol between SAP Cloud Deployment service and the content deployers.

They are used to indicate when the content deployment has reached a final state. Content deployers must include the markers in its logs. Then SAP Cloud Deployment service observes the application logs (STDOUT or STDERR) and decides if the content deployment has finished successfully or failed.

Success and failure markers are predefined and must not be changed by the end customer. They are represented by the module level parameters `success-marker` and `failure-marker`.

For instance, module type `com.sap.portal.site-content` has a `success-marker` `STDOUT:Deployment of site content to .* done.*`

## MTA Modelling

Below you can find the modelling for supporting the content deployment scenario for MTA module type `com.sap.portal.site-content`:

### Sample Code

```
_schema-version: "3.1"
ID: my-portal
version: "1.1"
modules:
  - name: site-content
    type: com.sap.portal.site-content
    path: site-content/
    parameters:
      buildpack: nodejs_buildpack // Predefined parameter
      no-route: true // Predefined parameter - avoid route mapping to the
                     content app. It will not serve any HTTP requests
      memory: 256M // Predefined parameter - used memory
      execute-app: true // Predefined parameter - flag that marks the app to
                         be used for simulated app execution
      success-marker: STDOUT:Deployment of site content to .* done.* // 
                     Predefined parameter - marker for success
      failure-marker: STDERR:Deployment of site content to .* failed.* // 
                     Predefined parameter - marker for failure
      stop-app: true // Predefined parameter - flag that specifies if the app
                     will be stopped after the actual deployment
```

```

check-deploy-id: true // Predefined parameter - flag that specifies
that certain and unique id will be checked for that content deployment
dependency-type: hard
health-check-type: none // Predefined parameter - skips the CF health-
check, so content app will not implement specific endpoint
requires:
  - name: sap-portal-services-client
  - name: portal-uaa

resources:
  - name: sap-portal-services-client // Provides content endpoint for the
actual content deployment. Content endpoint could be different based on the
account
    type: com.sap.portal.site-content
    parameters:
      config:
        siteId : 1234

  - name: portal-uaa // Used for secure communication between content app and
content endpoint
    type: com.sap.xs.uaa-space
    parameters:
      config-path: security/xs-security.json

```

## 4.1.8.1.9.7.4 CPI Content Deployment

This approach provides a mechanism for direct content deployment from the SAP Cloud Deployment service to content backend without the need for an intermediate Cloud Foundry application.

### ⓘ Note

This approach is only supported for MTA schema version 3.1 and higher.

This mechanism utilizes the CPI provided protocol. The end-to-end flow and MTA modelling are very similar to [Deploying Content with Generic Application Content Deployment \[page 677\]](#).

## Supported content type offerings

Cloud Foundry service offerings that currently support GACD are:

- API Management content - MTA module type `com.sap.api.management`
- CPI content – MTA module type `com.sap.integration`

## 4.1.8.1.10 Transporting Multitarget Applications in Cloud Foundry using CTS+

You can enable transport of SAP BTP applications and application content that is available as Multitarget Applications (MTA) using the Enhanced Change and Transport System (CTS+).

### Prerequisites

- Open the document [How to...Configure SAP BTP, Cloud Foundry environment for CTS](#).
- You have configured your SAP BTP subaccounts for transport with CTS+ as described in the document [How to...Configure SAP BTP, Cloud Foundry environment for CTS](#).
- The content that you want to transport can be made available as a Multitarget Application (MTA) archive as described in [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#).

### Context

You use the Change and Transport System (CTS) of ABAP to transport and deploy your applications running on SAP BTP in the form of MTAs, for example, from development to a test or production subaccount. Proceed as follows to be able to transport an SAP BTP application:

### Procedure

1. Package the application in a Multitarget Application (MTA) archive using the Archive Builder Tool as described in [Defining Multitarget Application Archives \[page 576\]](#).
2. Attach the MTA archive to a CTS+ transport request as described in [How to...Configure SAP BTP, Cloud Foundry environment for CTS](#).
3. Trigger the import of an SAP BTP application as described in [How to...Configure SAP BTP, Cloud Foundry environment for CTS](#).

### Related Information

[Resources on CTS+ !\[\]\(4d091e1b005f850230a1ec9ab1a9befb\_img.jpg\)](#)

[Setting up a CTS+ enabled transport landscape in SAP BTP !\[\]\(df9ed4542f593dc970c68822aaca3f0f\_img.jpg\)](#)

## 4.1.8.1.11 Frequently Asked Questions

### My MTA deployment failed. What could be done?

Check the [MTA Operation Failure](#) section.

### What to do when my application start-up/staging failed?

Check the [Application Start-up Failed](#) section.

### How can I find the logs of an application from a failed/succeeded deployment?

There are several ways to find the application logs:

- `OPERATION.LOG` – contains the whole deployment log.
- `<application-name>.log` – there is a separate file for each application. It holds the logs related to the application during staging and starting.

#### ⓘ Note

Note that the operation logs are kept for 3 days. Make sure that you download them before the retention period expires.

To download the logs, execute the following command:

```
cf dmol -i <process-id>
```

#### ↴ Sample Code

```
cf dmol -i cbe58aeb-0c40-4a3e-972d-82a499815745
```

### Whom to contact in case of problems with deployment?

When an error occurs during deployment, there is a message which indicates whether the problem is in one of the components of the Cloud Foundry Platform. In such cases, an incident to the corresponding component could be created.

#### ⚡ Example

"Controller operation failed: 400 Bad Request: Cannot bind application to service"

For more information, you can also check the [Troubleshooting \[page 694\]](#).

### How to check the state of my deployment?

Use the command `cf mta-ops` and find the id of the operation related to the desired deployment. There will be information about the status of the operation.

### How can I list the MTAs that I have deployed in my space?

Use the `cf mtas` command and locate the ID of the desired MTA. After locating the correct MTA ID, execute the command `cf mta <located-mta-id>` to get detailed information about the MTA with the provided ID.

## What can I do with my deployment?

- If the deployment fails, see [MTA Operation Failure](#).
- If the deployment finishes successfully, you can check the deployment logs.
- If the deployment is still running, you can abort or monitor it.

The **Abort** action is described in more details in the [MTA Operation Failure](#) section.

The **Monitor** action can be executed with the following command:

```
cf <operation> -i <operation-id> -a monitor. Example: cf deploy -i 12355 -a monitor
```

## How can I redeploy my MTA when a deployment is already running?

You can abort the currently running deployment, using the command `cf <operation> -i <operation-id> -a abort` (Example: `cf deploy -i 12353 -a abort`) or you can execute the command for starting the deployment by providing the option `-f` as described in the [deploy \[page 2457\]](#) section.

### ❖ Example

```
cf deploy <path-to-mtar>.mtar -f
```

## What are the size limits of an MTA?

4GB for the whole MTA and 1GB for a single module. For more information, see [Multitarget Applications for the Cloud Foundry Environment \[page 573\]](#).

## What to do in case of service creation failure?

See [Service Create Failure](#).

## How to make my deployment faster?

You can use a parallel deployment as described in section [Parallel Deployment \[page 673\]](#).

## How can I resolve problems with the MTA descriptor modeling?

For more information, see [Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 578\]](#).

## What is the deployment order of services, applications and other content?

Services are always deployed in parallel.

The applications and content deployment order is determined based on the `deployed-after` module parameter. If `deployed-after` is not used and parallel deployments are not enabled for the MTA, the `requires/provides` module sections define the order. For more information, see [Parallel Module Deployment \[page 672\]](#).

## How to reuse one backing service in two different MTAs?

If you want one MTA to “own” the lifecycle of the service and another to use it only as an “existing service”, use the `org.cloudfoundry.managed-service` and `org.cloudfoundry.existing-service` resource types respectively.

## How to define service instance and bindings parameters?

For more information, see [Service Binding Parameters \[page 643\]](#) as well as [Service Creation Parameters \[page 641\]](#).

## 4.1.8.1.12 Troubleshooting

This topic contains information that is essential for troubleshooting issues.

Before you start investigating the problem, download the logs for the current deployment. The logs contain the following structure of files:

- OPERATION.LOG - contains the whole deployment log
- <application-name>.log – there is a separate file for each application. It holds the logs related to the application during staging and starting

### ⓘ Note

Note that the operation logs are kept for 3 days. Make sure that you download them before the retention period expires

To download the logs, execute the following command:

```
cf dmol -i <process-id>
```

### ⚡ Example

```
cf dmol -i cbe58aeb-0c40-4a3e-972d-82a499815745
```

### Troubleshooting an MTA Operation Failure

To troubleshoot issues, see the list of problems that may occur during the Multitarget Application deployment that is available in [MTA Operation Failure \[page 694\]](#).

### Creating an Incident

If the list does not contain a solution for your issue, you can contact the service support team by reporting an incident through the [SAP Support Portal](#). Here is a list of what to provide in an incident:

1. Select the component *BC-CP-CF-DS* from the *Component* dropdown list.
2. In the description field, provide as much information as possible about the Cloud Foundry region, Cloud Foundry organization and space names and/or Global Unique Identifiers.
3. Provide information about scenario with the exact steps to be reproduced, expected result, actual result or the error that appears, relevant logs and screenshots. In all cases, the starting point for the investigation are the Multitarget Application operation logs.

## 4.1.8.1.12.1 MTA Operation Failure

There might be different reasons for an MTA deployment failure. Retrying the last failed step(s) of an operation is the first basic step you can take to recover from a failed deployment. This approach is useful for temporary issues with the underlying infrastructure but not helpful for configuration or functional problems.

To do this, enter the following command:

```
cf <deployment-action> -i <process-id> -a retry
```

### ❖ Example

```
cf undeploy -i cbe58aeb-0c40-4a3e-972d-82a499815745 -a retry
```

If retrying the last failed step(s) does not solve the problem, you can either try redeploying your MTA or troubleshoot the problem on your own.

To redeploy your MTA, proceed as follows:

1. Terminate the deployment process. To do this, enter the following command:

```
cf <deployment-action> -i <operation-id> -a abort
```

### ❖ Example

```
cf deploy -i cbe58aeb-0c40-4a3e-972d-82a499815745 -a abort
```

### ⓘ Note

This action will not roll back all applied changes to the previous stable version. It will allow the next deployments of that MTA to proceed without confirmation and the current process will end without any possibilities to retry it from the failed step-on.

2. Start a new deployment. See [Deploying to SAP BTP](#).

To troubleshoot the problem on your own, check the list of possible problems that may occur during the Multitarget Application deployment. If the problem cannot be categorized into any of the sections below, you need to [create an incident \[page 694\]](#).

## Service Create Failure

If the service create fails, an error message with the following format will be displayed:

```
...Error creating service "<service-name>" from offering "<service-offering>"  
and plan "<service-plan>": <cause-of-failure>
```

Such an error usually occurs due to a problem with the service provisioning infrastructure or the declared service creation parameters. To check if the issue is within the service provisioning infrastructure or the configuration outside the MTA, try creating the service using any of the listed methods in [Creating Service Instances](#)

### ⓘ Note

If any `create` parameters are used, make sure that they are the same as those that are used in the MTA. Note that the service creation parameters within the MTA may be located in different places, and their combination is required. For more information, see [Service Creation Parameters](#).

Depending on the result, proceed in one of the following ways:

- If the same error occurs, there is a problem with the service broker rather than the deploy service. Check the following:
  - Check your entitlement and quotas. Make sure that the quota is enough.
  - Check if the service parameters are correct.

If none of the above solves the problem, create an incident to the corresponding service broker component.

- If the above error is not reproduced, download the logs as described in [Troubleshooting \[page 694\]](#). Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Service Update Failure

If the service update fails, an error message with the following format will be displayed:

```
Error updating service "<service-name>" from offering "<service-offering>" and
plan "<service-plan>": <cause-of-failure>
```

Usually, such an error is produced when there is a problem with the services provisioning infrastructure. To check if the issue is within the service provisioning infrastructure or the configuration outside the MTA deployment, try updating the service using any of the listed methods in [Updating Service Instances](#).

### Note

If any update parameters are used, make sure that they are the same as those that are used in the MTA. Note that the service update parameters within the MTA may be located in different places, and their combination is important. For more information, see [Service Creation Parameters](#).

Depending on the result, proceed in one of the following ways:

- If the same error occurs, there is a problem with the service broker rather than the deploy service. Check the following:
  - Check your entitlement and quotas. Make sure that the quota is enough.
  - Check if the service parameters are correct.
 If none of the above solves the problem, create an incident to the corresponding service broker component.
- If the above error is not reproduced, download the logs as described in [Troubleshooting \[page 694\]](#). Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Service Delete Failure

If the service delete fails, an error message with the following format will be displayed:

```
Error deleting service "<service-name>" from offering "<service-offering>" and
plan "<service-plan>": <cause-of-failure>
```

Usually, such an error is produced when there is a problem with the services provisioning infrastructure. To check if there is a problem with the service itself, execute the following command:

```
cf delete-service <service-name> -f [-c PARAMETERS_AS_JSON]
```

Depending on the result, proceed in one of the following ways:

- If the same error occurs, there is a problem with the service broker rather than the deploy service. Create an incident to the corresponding service broker component.
- If the above error is not reproduced, download the logs as described in [Troubleshooting \[page 694\]](#). Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Application Binding to Service Failure

This error can occur when the services are created and the deployment is in phase, in which the application is being bound to the services. The error has the following format:

```
Could not bind application "<application-name>" to service "<service-name>":  
<cause-of-the-error>
```

Usually, this error happens when the service provider for the service fails to initiate the binding. The problem might also occur when the Cloud Foundry Cloud Controller component has internal issues.

Try to resolve the issue by unbinding/binding the application to the service manually, using the commands:

- **cf bind-service <application-name> <service-name> [ADDITIONAL OPTIONS]** - binds the application with the given name to service.
- **cf unbind-service <application-name> <service-name>** - unbinds the application with the given name from service.

If the above error is not reproduced, download the logs as described in [Troubleshooting \[page 694\]](#).

Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Application Content Upload Failure

The following error occurs:

```
Error uploading application <specific description>
```

To check if the issue is related to the MTA deployment, execute **cf push** instead of **cf deploy** using only the application with the problematic content.

If the command fails, there is a problem with the application, or the native Cloud Foundry, such as Cloud Controller or used buildpack(s). Troubleshoot your application and/or contact your application developer.

If the command is executed successfully, the problem is not related to the application, but there is a problem with the MTA deployment. You **must** download the logs as described in [Troubleshooting \[page 694\]](#). Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Application Tasks Execution Failure

To check if the issue is related to the MTA deployment, execute **cf run-task APPNAME "TASK"** instead of **cf deploy** using only the problematic task of the application.

If the command fails, there is a problem with the application, or the native Cloud Foundry, such as Cloud Controller. Troubleshoot your application and/or contact your application developer.

If the command is executed successfully, the problem is not related to the application or the task, but there is a problem with the MTA deployment. You **must** download the logs as described in [Troubleshooting \[page 694\]](#). Afterward, [create an incident \[page 694\]](#) or contact the SAP support.

## Applicatoin Starting Failure

In order to locate the problem in the application starting, the logs of the application should be checked.

Download the logs as described in [Troubleshooting \[page 694\]](#). There should be a dedicated file for the relevant application, which contains more traces for the starting and staging of the application. For instance, if the application is called `my-cf-app`, there would be a file called `my-cf-app.log`.

Normally, the failures during application starting indicate coding or configuration mistakes and self-troubleshooting is a must. Rarely, they might be related to problems in the Cloud Controller or the MTA deployment itself.

The following errors in the console output indicate problems during application starting:

```
Error starting application "<app_name>": Some instances have crashed. Check the logs of your application for more information.
```

```
Error starting application "<app_name>": Some instances are down. Check the logs of your application for more information.
```

### **Application Staging Failure**

In order to locate the problem in the application staging, the logs of the application should be checked.

Download the logs as described in [Troubleshooting \[page 694\]](#). There should be a dedicated file for the relevant application, which contains more traces for the starting and staging of the application. For instance, if the application is called `my-cf-app`, there would be a file called `my-cf-app.log`.

Normally, the failures during application staging indicate coding or configuration mistakes and self-troubleshooting is a must. Rarely, they might be related to problems in the Cloud Controller, buildpack or the MTA deployment itself.

The following error in the console output indicate problems during application staging:

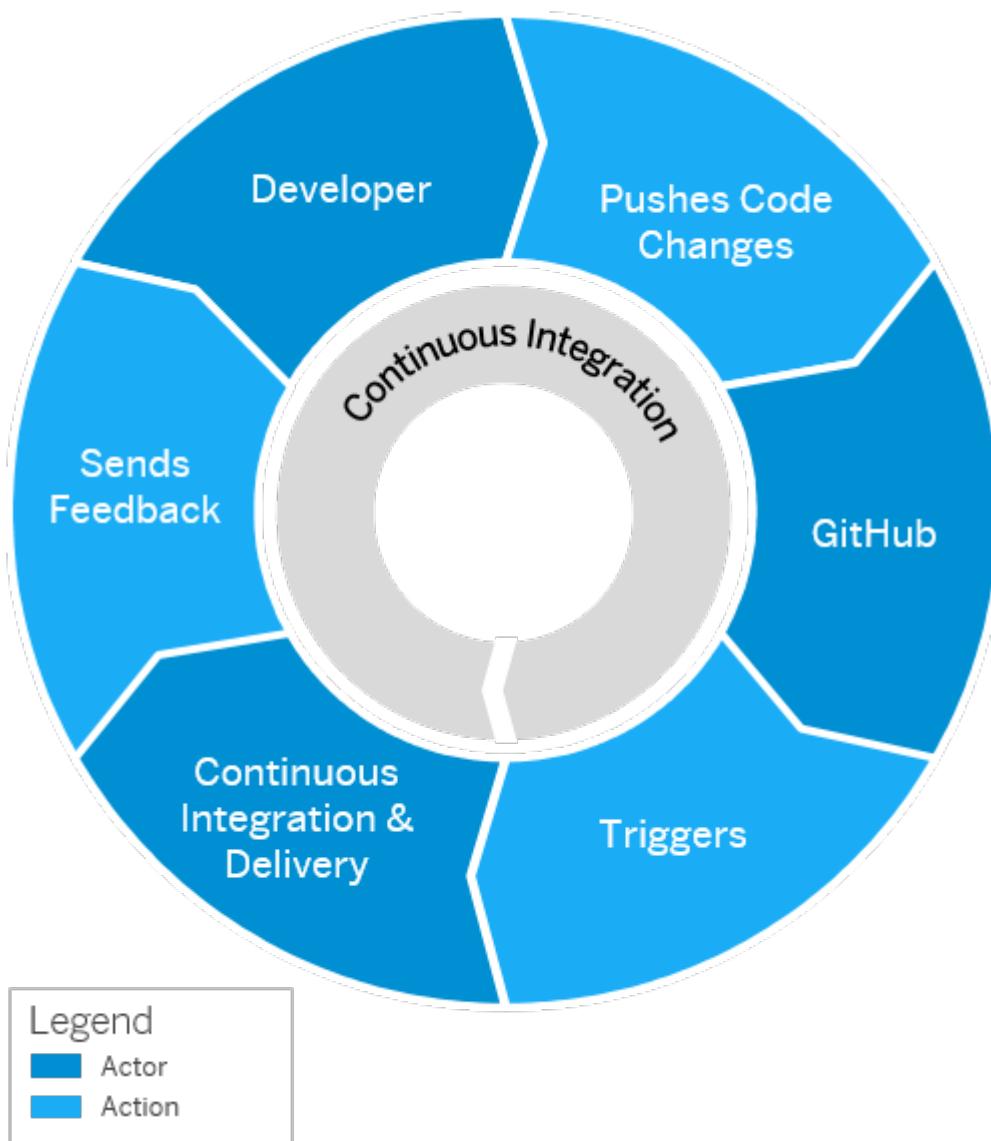
```
Error staging application "<app_name>": ...
```

## **4.1.8.2 Continuous Integration and Delivery**

Learn how to integrate CI/CD into your development with the SAP Cloud Application Programming Model (CAP).

### **4.1.8.2.1 What Are Continuous Integration and Delivery?**

**Continuous integration (CI)** describes a software development process in which various team members integrate their contributions frequently into a single main line. Before each integration, the changes are verified through builds and automated testing. Thereby, you can detect errors as quickly as possible and prevent integration problems before completing the development.



- [#unique\\_524/unique\\_524\\_Connect\\_42\\_subsection-im1 \[page 699\]](#)
- [#unique\\_524/unique\\_524\\_Connect\\_42\\_subsection-im2 \[page 700\]](#)
- [#unique\\_524/unique\\_524\\_Connect\\_42\\_subsection-im3 \[page 700\]](#)

→ Tip

Hover over each actor-action combination for a short description and click on them for more detailed information.

### Continuous Integration Basic Flow

The continuous integration basic flow comprises the following steps:

1. The developer writes code and pushes the code changes into a central source code management system (SCM).
2. The SCM triggers the continuous integration (CI) server.
3. The CI server runs automated builds and tests and sends feedback about their outcome to the developer.

### **Continuous Integration Basic Flow**

The continuous integration basic flow comprises the following steps:

1. The developer writes code and pushes the code changes into a central source code management system (SCM).
2. **The SCM triggers the continuous integration (CI) server.**
3. The CI server runs automated builds and tests and sends feedback about their outcome to the developer.

### **Continuous Integration Basic Flow**

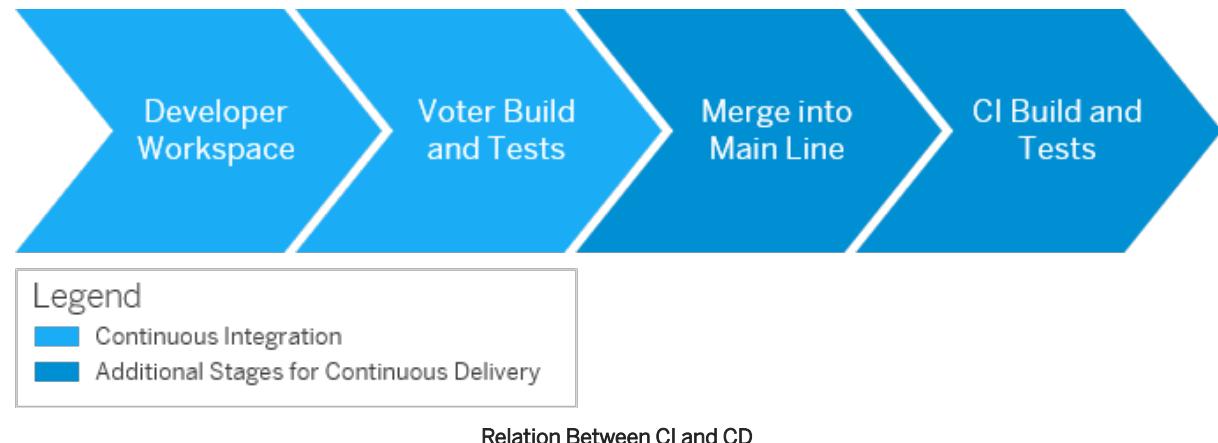
The continuous integration basic flow comprises the following steps:

1. The developer writes code and pushes the code changes into a central source code management system (SCM).
2. The SCM triggers the continuous integration (CI) server.
3. **The CI server runs automated builds and tests and sends feedback about their outcome to the developer.**

As you can see from the graphic, the basic flow for continuous integration is a cycle: As soon as the CI server has sent its feedback to the developer, the flow starts over again. The developer either corrects his or her previous code change, which must then be built and tested again, or starts working on an entirely new one.

The **continuous delivery (CD)** concept expands on the one of continuous integration. It adds the aspect that any change that has successfully passed the tests is immediately ready to be deployed to production, both from a technical and a qualitative point of view.

The following graphic shows the relation between continuous integration and continuous delivery:



The continuous delivery process makes sure that the most current version of the software product is successfully built, tested, and provided in a shippable format. Based on the release decision by the development team or delivery manager, it can be shipped to customers or deployed to production at any time.

For more information about the concepts and principles of continuous integration and delivery, see the [Continuous Integration and Delivery Introduction Guide](#).

## 4.1.8.2.2 CI/CD Solutions by SAP for the SAP Cloud Application Programming Model

SAP offers two different solutions that help you apply CI/CD in your development with the SAP Cloud Application Programming Model (CAP):

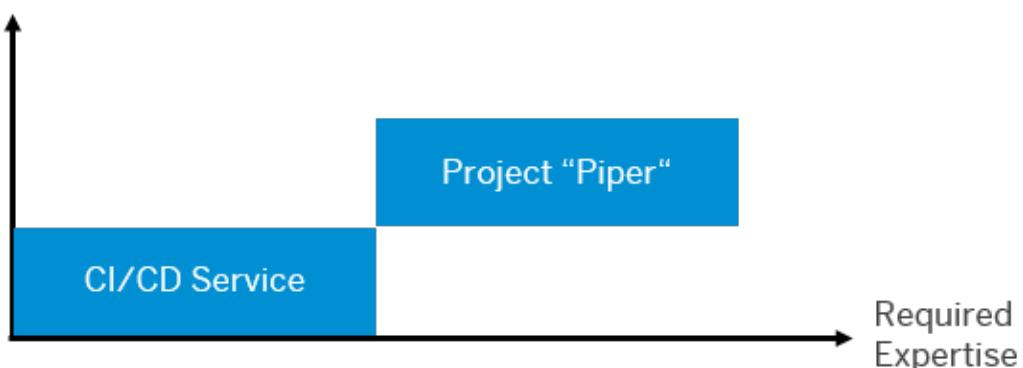
- **SAP Continuous Integration and Delivery** is a service on SAP BTP, which lets you configure and run predefined CI/CD pipelines that test, build, and deploy your code changes.
- **Project “Piper”** is an open-source project that provides preconfigured Jenkins pipelines, which you can use in your own Jenkins infrastructure and adapt according to your needs, if necessary. It consists of a [shared library](#), which contains the description of steps, scenarios, and utilities required to use Jenkins pipelines, and a [set of Docker images](#) that can be used to implement best practice processes.

Both solutions offer in their level of flexibility and expertise required for setup and configuration:

### ⓘ Note

In the following image, click on one of the blue boxes to navigate to the corresponding solution.

### Flexibility



- <https://sap.github.io/jenkins-library/> ↗ [https://sap.github.io/jenkins-library/]
- [https://help.sap.com/viewer/product/CONTINUOUS\\_DELIVERY/Cloud/en-US](https://help.sap.com/viewer/product/CONTINUOUS_DELIVERY/Cloud/en-US) [https://help.sap.com/viewer/product/CONTINUOUS\_DELIVERY/Cloud/en-US]

As you don't need to care about the underlying infrastructure, SAP Continuous Integration and Delivery requires the least expertise in CI/CD. Therefore, however, its flexibility is limited. Project “Piper” Docker images can be used out-of-the-box. However, this offering depends on Jenkins as underlying CI/CD tool.

For more information about the CI/CD solutions by SAP, see [SAP Solutions for Continuous Integration and Delivery](#).

The following links guide you to the SAP Cloud Application Programming Model (CAP) sections of our CI/CD solutions:

CI/CD Solution	CAP Section	Description
SAP Continuous Integration and Delivery	<a href="#">Configure an SAP Cloud Application Programming Model Job in Your Repository</a>	Configure a CI/CD pipeline for the development of applications that follow the SAP Cloud Application Programming Model in the Cloud Foundry environment.
Project "Piper"	<a href="#">Build and Deploy SAP Cloud Application Programming Model Applications</a>	Set up a CI/CD Pipeline for an SAP Cloud Application Programming Model (CAP) project.

### 4.1.8.3 Deploy Business Applications in the Cloud Foundry Environment

When an application for the Cloud Foundry environment resides in a folder on your local machine, you can deploy it and start it by executing the command line interface (CLI) command `push`. To deploy business applications bundled in a multitarget application archive, you have to use the command `deploy-mta`.

- For more information about developing and deploying applications in the Cloud Foundry environment, see <http://docs.cloudfoundry.org/devguide/deploy-apps/deploy-app.html>.
- For more information developing Multitarget Applications, see [Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#). See [Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#) for information about MTA deployment.

#### Related Information

[Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

[Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)

[Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#)

## 4.1.8.4 Deploy Docker Images in the Cloud Foundry Environment

The high-level steps to application deployment using Docker images.

### Prerequisites

- You are aware how `cf push` works when you deploy a regular Cloud Foundry application.
- You need a higher degree of freedom and know that this freedom comes with higher responsibility for application operation.
- Your Docker image resides in a highly available container registry.

### Context

A Docker image deployed to the Cloud Foundry environment has to adhere to the same boundary conditions as regular Cloud Foundry applications. Cloud Foundry applications deployed as Docker images are running in Linux containers with user namespaces enabled, which prevents certain functionality like mounting FuseFS devices. Another difference is the usage of buildpacks. Docker images can't be used in combination with buildpacks. If you want to use certain functionality provided by the buildpack, you have to build it into your Docker image.

Compared to regular Cloud Foundry applications, using a Docker image means more effort for you. If the following factors are fulfilled, using Docker on Cloud Foundry might be a viable option.

1. Managing a whole cluster on Kubernetes is too much overhead for your scenario.
2. The usual approach doesn't let you realize what you want to achieve.

Here are some features of the Cloud Foundry environment you have to implement yourself in your Docker image. The comparison is made with the SAP Java buildpack.

Supported with SAP Java buildpack	Description
Dynatrace	Enable applications for Dynatrace based monitoring.
XSUAA	Automatic validation of OAuth token and mapping of OAuth scopes to JEE roles, if <code>auth-method</code> is set to XSUAA in <code>web.xml</code> .
Dynamic Logging	Set of log level via CLI without restart.
Switch to debug via CLI	
OS & JVM management	Consume newer versions (especially patches) of the underlying OS and JVM regularly..

## Procedure

1. Your Docker image adheres to the 12 factor principles.

<https://12factor.net/> ↗

2. You only use supported features in the SAP BTP, Cloud Foundry environment, see [Cloud Foundry Environment \[page 49\]](#)

3. Read what the open source documentation says about deploying an application with Docker.

Pay special attention to the mentioned requirements. [Deploy an App with Docker](#) ↗

Requires traffic routing to the lowest numbered port. [How Diego Runs and Monitors Processes](#) ↗

4. Use `cf push` with the `--docker-image` parameter.

This is an example if you want to push from Docker Hub:

```
cf push <APP-NAME> --docker-image <REPO>/<IMAGE>:<TAG>
```

Also pay attention to the section on private registries and Google Container Registry. [Push a Docker Image From Docker Hub](#) ↗

5. Operate and patch your Docker images.

Docker images contain everything that is necessary to run a process, including the operating system and libraries into one binary blob. Consequently, staging a Docker image in Cloud Foundry does not provide the same separation of droplet and stack that is available for a Cloud Foundry application staged using a buildpack.

If there is an operating system and library security vulnerability, the developer is responsible to `cf push` a new version of the Docker images . The developer using the platform has the responsibility for watching these security vulnerabilities and for reacting accordingly.

For scale or restart operations, the image is pulled again from the container registry. The container registry has to be highly available and reachable from network side for proper operations.

## Results

Was this topic helpful? Did you miss something? Let us know and use the feedback function ↗ (feedback).

## 4.2 Development in the ABAP Environment

Learn more about developing applications in the ABAP environment.

### Overview

The ABAP environment is a platform as a service that allows you to extend existing ABAP-based applications and develop ABAP cloud apps decoupled from the digital core. You can leverage your ABAP know-how in the cloud and reuse existing ABAP assets by writing your source code with ABAP Development Tools for Eclipse.

The ABAP environment enables you to **expose**:

- OData services. See [ABAP RESTful Application Programming Model](#).
- HTTP services. See [Working with the HTTP Service Editor](#).
- SQL services. See [Data Federation Using the SQL Service](#).
- RFC function modules. See [Providing an RFC Service](#).
- SOAP provider model. See [Providing a SOAP Service](#).

With your ABAP applications, you can **consume**:

- HTTP services (HTTP client). See [Outbound HTTP Administration Tasks](#).
- OData services (service consumption model). See [Developing a UI Service with Access to a Remote Service](#).
- Remote Function Calls (RFC). See [Outbound RFC Administration Tasks](#).
- On-premise systems via Cloud Connector. See [Integrating On-Premise Systems \[page 1184\]](#).
- SOAP-based Web services. See [Outbound SOAP Administration Tasks](#).

### Technical Limits and Boundary Conditions for ABAP Applications

For elasticity, operability and protection reasons, the runtime of API or UI requests to ABAP applications is limited. After 10 minutes, running requests are automatically canceled. Tasks requiring longer runtimes have to be executed in [Application Jobs](#), as background processes via the background processing framework (bgPF), or need to be split into smaller work packages. For an optimal user experience, all ABAP applications, including application jobs, have to be prepared to be interrupted and restarted even earlier than the maximum runtime. This allows to provide a continuous application service, unimpacted from scaling operations, maintenance activities or even infrastructure failures.

### Development Resources

[ABAP RESTful Application Programming Model](#)

[ABAP Development Tools for Eclipse: User Guide](#)

[ABAP CDS Development User Guide](#)

[ABAP Keyword Documentation \[page 711\]](#)

[ABAP Lifecycle Management \[page 715\]](#)

[Connect to the ABAP System \[page 708\]](#)

[Providing an HTTP Service](#)

[Integrating On-Premise Systems \[page 1184\]](#)

[Supported Protocols and Authentication Methods](#)

[Inbound Communication](#)

[Working with abapGit \[page 720\]](#)

## Related Information

[Eclipse Tool for the ABAP Environment](#)

[Learning Journey](#)

[Manage Software Components \[page 2805\]](#)

[Video Tutorials !\[\]\(2d2c04895a901b449cdd5bf1b56344cf\_img.jpg\)](#)

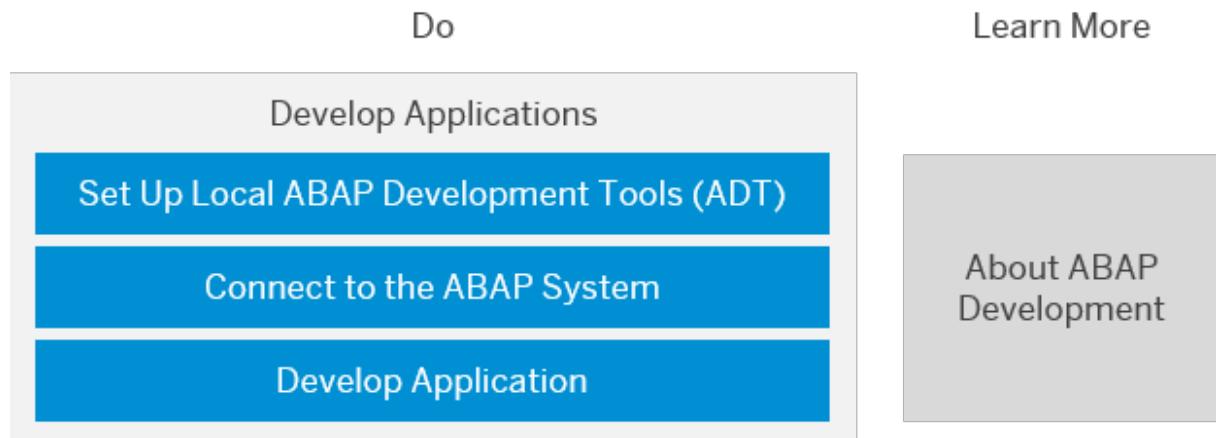
### 4.2.1 Getting Started as a Developer in the ABAP Environment

Learn about your first steps to get started as a developer in the ABAP environment.

#### Note

This documentation informs you about the first steps as a developer and assumes that the necessary steps for setting up the ABAP environment are done. For more information about how to set up the ABAP environment and getting started as an administrator, see [Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#).

## Developing Applications: Overview



- <https://tools.hana.ondemand.com/#abap> [https://tools.hana.ondemand.com/#abap]
  - Connect to the ABAP System [page 708]
  - <https://developers.sap.com/tutorials/abap-environment-gcts.html#b1afe95c-9acb-4f75-8f80-1abe93cb524f> [https://developers.sap.com/tutorials/abap-environment-gcts.html#b1afe95c-9acb-4f75-8f80-1abe93cb524f]
  - <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html]
  - <https://help.sap.com/viewer/DRAFT/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/59656c2f858748fe976456248d390c5c.html> [https://help.sap.com/viewer/DRAFT/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/59656c2f858748fe976456248d390c5c.html]
  - <https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html> [https://help.sap.com/viewer/65de2977205c403bbc107264b8eccf4b/Cloud/en-US/31367ef6c3e947059e0d7c1cbfcaae93.html]
1. Download and install ABAP Development Tools (ADT) from <https://tools.hana.ondemand.com/#abap>. See [Video Tutorial: Configure ABAP Development Tools](#).

### ⓘ Note

SAP GUI is not supported in the ABAP environment. You can only use ADT as your development environment.

2. Create an ABAP cloud project with ADT to connect to the ABAP system in the ABAP environment (see [Connect to the ABAP System](#) [page 708]).
3. Develop your application (see [Development in the ABAP Environment](#)). To learn more about how to develop applications, see [Tutorial: Create Your First ABAP Console Application](#) and [Video Tutorial: Create Application](#).

## More Information About Development in the ABAP Environment

For more information about ABAP development, see [Development in the ABAP Environment](#).

### 4.2.1.1 Connect to the ABAP System

Set up your ABAP cloud project to connect to the ABAP system.

#### Prerequisites

- You are assigned the developer role. See [Assigning the ABAP Developer User to the ABAP Developer Role \[page 192\]](#) and [Creating Service Keys](#).

##### ⓘ Note

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

- You have downloaded and installed the front-end components of [ABAP Development Tools \(ADT\)](#). See [Video Tutorial: Configure ABAP Development Tools](#).
- You are a member of the relevant space in the Cloud Foundry environment, see [Add Space Members Using the Cockpit](#), or you have a service key in JSON format, see [Creating Service Keys](#).

#### Procedure

1. Open Eclipse and select **File > New > ABAP Cloud Project** from the menu.
2. To establish a service instance connection, you have the following options:
  - a. **[Default] Service key provided by Cloud Foundry environment:**

In the [New ABAP Cloud Project](#) wizard, on the [System Connection to SAP BTP ABAP Environment](#) page, select the [Use Cloud Foundry Environment](#) radio button, and choose [Next](#).

On the [Connection Settings](#) page, select [Europe \(Frankfurt\)](#) as your region, enter your email address and password, and confirm with [Next](#).

On the [Connection Settings to SAP BTP Cloud Foundry Environment](#) page, select your service instance details.
  - b. **Service key in JSON format:**

In the [New ABAP Cloud Project](#) wizard, on the [System Connection to SAP BTP ABAP Environment](#) page, select the [Service Key](#) radio button, and choose [Next](#).

On the *System Connection Using a Service Key* page, paste your existing service key from the clipboard into the *Service Key in JSON Format* text box, or choose *Import* to import a text file containing your service key.

#### ⓘ Note

If you're an SAP Universal ID user, you can only use this option to establish a service instance connection.

3. Select *Next*.
4. To log on to the service instance on the *System Connection Using a Service Key* page, you have the following options:
  - a. **Using the integrated browser:** Enter your email address and password.

#### ⓘ Note

Authentication is performed in the integrated browser. Single Sign On is not supported. Tools, such as password managers, are not supported for this logon option.

- b. **Using the default browser:** Select the *Log on with Browser* button.
- c. **Using another browser:** Choose the *Copy Logon URL* button.

The URL is copied to the clipboard of your computer.

5. To connect to your service instance, select *Log On*.

On the *Service Instance Connection* page, the following connection settings are displayed:

- **Service Instance URL:** URL of the server instance where the ABAP system is operated
- **Email:** ID of the user who is authorized in the configured identity provider for accessing the ABAP system
- **User ID:** ID of the user who is assigned to the email
- **SAP System ID:** Name of the ABAP system
- **Client:** ID of the logon client
- **Language:** Abbreviation of the logon language

#### ⓘ Note

Currently, English is the only logon language.

6. Select *Next*.

The *Project Name and Favorite Packages* page is opened.

7. [Optional:] If you want to change the name of the project, enter a new name in the *Project Name* field.

#### ⓘ Note

When you create the project, the `ZLOCAL` ABAP package is added by default to your *Favorite Packages*. This ABAP package including all subpackages contains all local objects from every user of the ABAP system.

To add further ABAP packages to your *Favorite Packages*, choose *Add...* and enter the name of the package in the corresponding input field. Note that this package must be available in the system.

To group your projects, you can add working sets. To do so, choose *New...* in the *Working sets* section and select the relevant projects.

8. To create your ABAP cloud project, select *Finish*.

## Results

You have created an ABAP cloud project that is added to the *Project Explorer*.

### ⓘ Note

To verify your result, expand the first level of the project structure. Make sure that the following nodes are included:

- *Favorite Packages*: Contains the local packages and the packages that you have added to your Favorites.
- *Released Objects*: Contains all released SAP objects that are available to (re)use.

## Related Information

[Video Tutorial: Create ABAP Cloud Project](#) ↗

[Video Tutorial: Create Package & Persistence](#) ↗

[Tutorial: Create Your First ABAP Console Application](#) ↗

[ABAP Cloud Projects](#)

## 4.2.2 ABAP Development Tools for Eclipse: User Guides

Get an overview of all available user guides for ABAP development tools for Eclipse.

### [ABAP Development Tools for Eclipse: User Guide](#)

This guide describes the functionality and usage of the possibilities within the ABAP development tools for Eclipse. It focuses on use cases for creating, editing, testing, debugging, and profiling development objects.

### [ABAP CDS Development Tools: User Guide](#)

This guide describes the functionality and usage of tools for Core Data Services (CDS) in the ABAP environment. It focuses on use cases for creating, editing, testing, and analyzing ABAP CDS entities.

## 4.2.3 ABAP Keyword Documentation

The ABAP keyword documentation describes the syntax and meaning of the keywords of the ABAP language and its object-oriented part – ABAP objects.

### Context

The ABAP keyword documentation provides you with context-sensitive information for your ABAP source code.

### Accessing ABAP Language Help

To access the ABAP language help from the source code editor, position your cursor on an ABAP statement for which you need help, and press **F1**. The language help is displayed in a separate window.

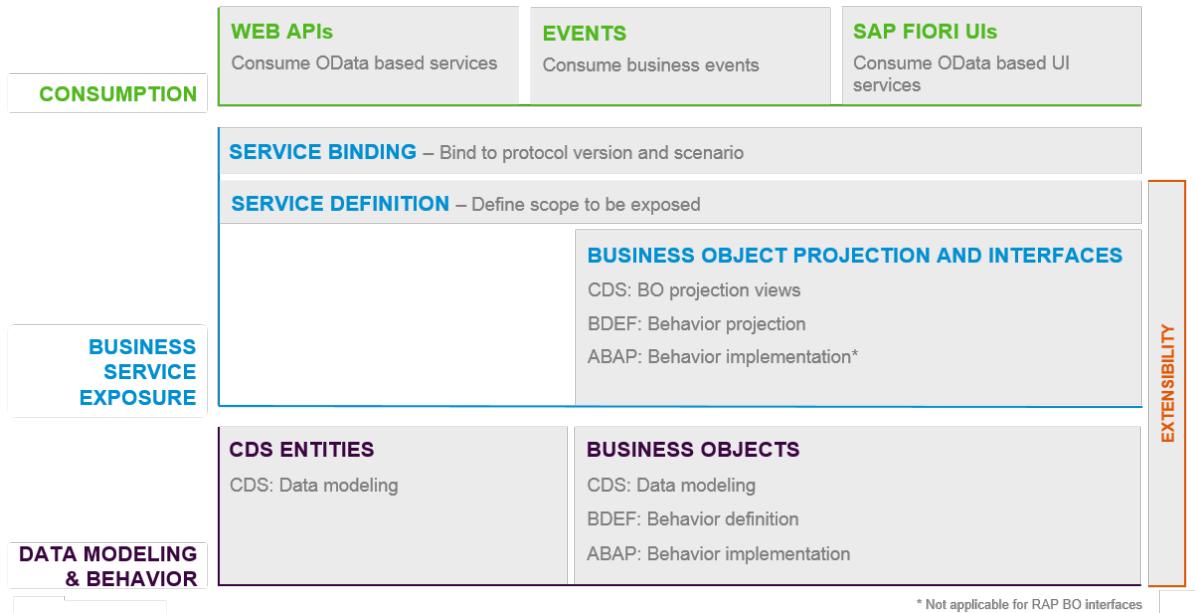
To view the entire ABAP keyword documentation, see [ABAP - Keyword Documentation](#).

## 4.2.4 ABAP RESTful Application Programming Model

The [ABAP RESTful Application Programming Model](#) defines the architecture for efficient end-to-end development of intrinsically SAP HANA-optimized OData services (such as SAP Fiori apps) in the ABAP environment. It supports the development of all types of Fiori applications as well as publishing Web APIs. It is based on technologies and frameworks such as Core Data Services (CDS) for defining semantically rich data models and a service model infrastructure for creating OData services with bindings to an OData protocol and ABAP-based application services for custom logic and SAPUI5-based user interfaces.

[Architecture Overview](#)

## RAP - The big picture



- [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/cc80e974d128426d9d876d5a8fc9e4aa.html#loiocc80e974d128426d9d876d5a8fc9e4aa\\_\\_Design\\_Time](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/cc80e974d128426d9d876d5a8fc9e4aa.html#loiocc80e974d128426d9d876d5a8fc9e4aa__Design_Time) [[https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/cc80e974d128426d9d876d5a8fc9e4aa.html#loiocc80e974d128426d9d876d5a8fc9e4aa\\_\\_Design\\_Time](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/cc80e974d128426d9d876d5a8fc9e4aa.html#loiocc80e974d128426d9d876d5a8fc9e4aa__Design_Time)]
- [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiocc80e974d128426d9d876d5a8fc9e4aa\\_\\_Web\\_API](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiocc80e974d128426d9d876d5a8fc9e4aa__Web_API) [[https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiocc80e974d128426d9d876d5a8fc9e4aa\\_\\_Web\\_API](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiocc80e974d128426d9d876d5a8fc9e4aa__Web_API)]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/b58a3c27df4e406f9335d4b346f6be04.html> [<https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/b58a3c27df4e406f9335d4b346f6be04.html>]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/b09e4d53bfca4544a9f8910bcc2cd9d6.html> [<https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/b09e4d53bfca4544a9f8910bcc2cd9d6.html>]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/6e7a10d30b74412a9482a80b0b88e005.html> [<https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/6e7a10d30b74412a9482a80b0b88e005.html>]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/a3ff9dcdb25a4f1a9408422b8ba5fa00.html> [<https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/a3ff9dcdb25a4f1a9408422b8ba5fa00.html>]
- <https://help.sap.com/docs/btp/sap-abap-restful-application-programming-model/query?version=Cloud> [<https://help.sap.com/docs/btp/sap-abap-restful-application-programming-model/query?version=Cloud>]

- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html> [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/1913ad9f52e64ab5858df00a8d20c4d6.html> [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/1913ad9f52e64ab5858df00a8d20c4d6.html]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/c4aaecac48294ef1a39ef13de0706a4b.html> [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/c4aaecac48294ef1a39ef13de0706a4b.html]
- <https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/0b925bc556d4491aad395b21ec2566ff.html> [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/0b925bc556d4491aad395b21ec2566ff.html]

## Related Information

Tutorial: Get to Know the ABAP RESTful Application Programming Model 

### 4.2.5 Assuring Quality of ABAP Code

This is an overview of ABAP quality tools that help you to keep your custom code clean and performant.

The two most important quality tools for your custom ABAP code are:

- [ABAP Unit](#) for dynamic tests
- [ABAP Test Cockpit \(ATC\)](#) for static code analysis

## ABAP Unit

Writing ABAP Unit tests is the way to provide high-quality software, which can be easily evolved over time without introducing regressions. In methodologies, like extreme programming and test-driven development, the role of unit testing is even more important.

ABAP unit is the state-of-the-art unit testing framework for ABAP. It's embedded into the ABAP programming language, which supports you in writing unit tests. In the ABAP Development Tools for Eclipse (ADT), you have various possibilities to execute the unit tests and to evaluate the results concerning functional correctness and code coverage.

We recommend measuring the test coverage of all objects in a transport before releasing the transport. A high coverage (above 80%, for example) and a wide variety of test methods help to identify issues and make the code more robust.

For more information, see [Unit Testing with ABAP Unit](#).

## ABAP Test Cockpit (ATC)

ATC is the tool of choice for all kinds of static code analysis in ABAP. As an ABAP developer, you can use ATC in the ABAP Development Tools for Eclipse to find potential bugs already during the development phase.

In particular, you can check ABAP development objects for many types of problems, including syntax errors, potential problems in performance, suboptimal or potentially faulty programming, adherence to standards, errors in ABAP Unit testing, among others.

For more information, see [Checking Quality of ABAP Code with ATC](#).

### ATC Check Variant

The basic configuration of the ATC is the so-called ATC check variant, which includes the set of checks that should be executed. The default variant is set to ABAP\_CLOUD\_DEVELOPMENT\_DEFAULT. It includes all recommended checks to fulfill the ABAP cloud code quality standard incl. the execution of ABAP Unit tests.

You can create a copy of this variant and add additional checks. You can also add custom created checks to your variant.

For more information, see [Working with ATC Checks](#).

### Transports

The basic quality gate is the release of transport tasks and transport requests.

For findings with priority level 1 and 2, the release of tasks and requests is blocked by the system in the default setting. For findings with priority level 3, the developer only gets a notification. This default setting can be individually sharpened based on your needs.

For more information, see [Working with ATC During Transport Release](#).

### Test Runs

You can either set up scheduled automated test runs for the ATC, or trigger it manually via ADT.

For more information, see [Working with ATC During Development](#).

### ATC Exemptions

Sometimes ATC findings can't be corrected. For example, it's possible that a correction is planned for an upcoming development cycle or only a test program is affected. It could also happen, that checks return false positives.

In this case, you can still process the issue by requesting an exemption. Once approved, it masks an ATC error or warning message. The finding then isn't displayed as an open issue in the ATC results anymore.

In the ABAP environment, exemptions are transportable objects of the type CHKE. The full workflow for the request and approval of exemption is integrated in ADT.

For more information, see: [Working with ATC Exemptions](#).

### ABAP Test Cockpit Configurator App

You can use the ATC Configuration app to set up the default variant. You can also change the priorities of check messages, set priority levels that block or interrupt transport releases, and set the handling of pseudo comments and pragmas.

For more information, see [ABAP Test Cockpit Configurator](#).

## 4.2.6 ABAP Lifecycle Management

This chapter helps you to plan and set up your landscape and lifecycle management by describing how software components can be used and which possibilities you have for transportation of development objects and business configuration.

### 4.2.6.1 Basic Concepts and Terms

To better understand how software lifecycle management with gCTS in the ABAP environment can be implemented, let's start off with some basic concepts and terms.

#### Classical Git

##### Git

Git is a distributed version-control system.

Distributed version control is a form in which development objects and all their versions and metadata are mirrored from a central place to every developer's computer. The developer can switch locally and offline between different versions and change all of them before updating the central originals later.

##### Repository

A repository is a collection of objects, their directory structure, and metadata, for example, a historical record of changes in the repository, a set of commit objects, and a set of references to commit objects.

The central place that all objects are mirrored from and saved to, after having finished changes on a developer's computer, is the so-called remote repository. Every developer's computer holds a so-called local repository to work on.

##### Branch

Git repository branches can be used to control the flow of changes through the test and production landscape. They are used to separate changes from each other, which shall or might not be delivered together (i.e. development vs. correction, feature A vs. feature B). A branch is created on the current state of another branch, the parent branch, and reflected by a new copy of all included objects, that is, the commit history of the parent branch. Depending on which branch is supposed to be changed, the corresponding copy is worked on.

##### Pull

Pulling means getting the code that is currently in the remote repository into the local repository.

##### Push

Pushing means getting the code that is currently in the local repository into the remote repository.

##### Commit

A commit records the state and content changes of a file. It bundles multiple changes and saves it to the local repository. When you perform a git push, all commits are transferred to the remote git.

## Check Out

Checking out means that, if several branches are available, the one that is selected is added to the working tree. The working tree consists of the objects of the branch you are currently working on. It contains the state of these objects since they were last pulled and the changes that were made locally but are not yet pushed to the central remote repository. This checked out branch/working tree is what is displayed in the application to edit the objects.

## ABAP and ABAP with gCTS

### ABAP System/Instance

SAP Business Technology Platform offers a service called ABAP System (abap). If you want to develop with ABAP in the cloud, you have to create an instance of that service, see [Create an ABAP System](#). This service instance provides the ABAP system that you are developing in. The terms ABAP system and ABAP instance can and are often used synonymously, however, we refer to the term ABAP system.

### Transport Request

A transport request records all the changes in your ABAP development system. Once a transport request is released, the changes are pushed into your central Git repository, to be more precise a branch of it, which is by default the main branch, in the cloud as a commit represented by a commit ID. During the import (pull or checkout), the delta of the Git repository branch content between the ABAP system and central Git repository is imported.

#### ⓘ Note

Transport requests cannot be imported selectively into subsequent ABAP systems but as part of software component branches their changes were added to.

### Release

A release, for example with the format YYYY-<nn>, is a set of changes that are tested successfully in your quality ABAP system(s) and imported into your production ABAP systems after a release decision. It is represented by a release branch in the Git repository. Once the release decision has been made, you have to create and pull (import) the new release branch into the production ABAP system.

## 4.2.6.2 Transport Mechanisms

There are two git-based transport mechanisms in the ABAP environment.

**Git-based CTS (gCTS)** is the evolution of the classical Change and Transport Management System (CTS). It is the recommended approach for transporting objects between ABAP systems in your global account. It offers built-in and easy to use functionalities provided by the [Manage Software Components](#) app in your SAP Fiori launchpad.

**abapGit** is an open-source Git client that allows you to import existing code into your ABAP system. You should use it for the following use cases:

- Migrate on-premise code to the cloud. See [Use abapGit to Transform ABAP Source Code to the Cloud](#).
- Transfer your code from the cloud to on premise. See [Transfer Your ABAP Source Code With SAP BTP ABAP Environment via abapGit](#).
- Export your code when the development ABAP system is decommissioned.
- Transfer your code from one cloud to another to share it with others, for example as open source, or from a partner to a customer account.
- Implement mechanisms for distributed development and testing in dedicated development/test ABAP systems. This can be useful for special projects, such as proof of concepts or features that are dependent on regular development of a solution but shall run independent from its lifecycle.
- To learn more about abapGit, see [Working with abapGit \[page 720\]](#).

For a quick start on transportation, see [Transport a Software Component Between Two ABAP Systems](#). For more information about the according delivery process, see [Delivery via Add-On or gCTS](#).

### 4.2.6.2.1 Software Components

Your development, including transportation, is organized and managed in so-called software components. A software component is designed to contain all coding and development objects of at least one application and should be runnable itself.

#### ❖ Example

If you think of an application for bonus management, this is how your application could be structured:

- `Z_BONUS_MANAGEMENT` (top-level package, corresponds to the software component and is created automatically, type `Structure`)
  - `Z_BONUS_CALCULATION` (sub-package, type `Development`)
  - `Z_BONUS_CONFIGURATION` (sub-package, type `Development`)

#### → Tip

Your development can also be loosely coupled in different software components.

#### ⓘ Note

You must not create objects in the structure package. You have to create a sub-package of type `Development` first to start developing.

#### ⚠ Restriction

- Objects of one software component cannot be used in another software component by default because software components provide their functionality via explicitly released APIs to other software components. This means you must set the API state of the object to `Released` if you want to use an object from another software component. See [Released APIs](#), [Finding Released APIs](#) and [Deprecated Objects](#).
- Software components should not have cyclic dependencies (objects in software component A use objects in software component B and vice versa). If you create dependencies between software

components, we recommend that you do this in a layered way, for example, for grouping basic reuse functions in a dedicated software component.

- You can't move development objects from one software component to another. Thus, the introduction of software components should be planned carefully.
- You can't move objects between packages that reside in different transportable software components.
- You can't move objects between packages that reside in a transportable software component to `ZLOCAL`.

For more information about moving development objects into ABAP packages, see [Changing the Package Assignment of Development Objects](#).

- You can use the same ABAP systems regarding development, testing, and production (your system landscape) for all your software components. Lifecycle processes for software components can even be independent from each other, as long as there are no development dependencies.
- gCTS only supports certain object types. For a complete list of the restricted object types, see SAP note [2888887](#).
- gCTS repositories are currently always stored on the AWS data center **Europe - Frankfurt EU Access**, independent of the chosen hyperscaler and the data center where the ABAP environment instance was created.

You create your software components in the development system:

- The software component `ZLOCAL` is available by default. It serves a similar role like `$TMP` in an on-premise system.
- Create your software components with the SAP Fiori app [Manage Software Components](#) (business catalog `Lifecycle Management - Software Components` `SAP_A4C_BC_MSCL_PC`). Afterwards, pull the software component into the ABAP system to start developing in it.
- You can use an ABAP namespace in the ABAP environment. If you have registered a namespace at SAP, it is automatically provided during provisioning. For more information on namespaces, see SAP note [105132](#) on how to reserve a namespace and ONE Support Launchpad [Namespace Application](#). A developer key and repair key are created and assigned automatically by the ABAP system. In the namespace application, you can search for your key assignments by filtering the installation number (`CLOUDSYSTEM`).
- If you want to transport business configuration content across ABAP systems, create a software component. You have to decide which type you want to use for transporting business configuration:
  - You can create a software component of type `Business Configuration` and transport all your configuration via this software component. If a software component of type `Business Configuration` is available in a tenant, applications can automatically select or create a customizing transport request for this software component.

### Note

You can only import one software component of type `Business Configuration` per ABAP system.

- You can transport the configuration using the software components that you also use for your development objects, which is typically the same software component for tables and table content. This is advisable if you run multiple projects on the same landscape that are well separated and have decoupled time schedules.

See [Transport of Business Configuration \[page 760\]](#).

### ⓘ Note

In your system you will see all software components linked to the same global account. Region or infrastructure provider do not matter. You only have to ensure that all systems are created within the same global account.

## 4.2.6.3 ABAP Environment Specifics

gCTS and the SAP Fiori app Manage Software Components provide a simplified git UI adapted to the possibilities of ABAP. This means:

- There is a remote Git repository for each software component managed by SAP in the cloud. Using the SAP Fiori app Manage Software Components, a software component can be created from any ABAP system in the same global account. Alongside with the software component creation, the corresponding remote repository is created. Afterwards, the software component needs to be cloned and pulled into the ABAP system. With this step, a corresponding structure package is created in the ABAP system that acts as a local repository so that you can start developing (software component = repository in gCTS).
- The creation of a Git repository branch is also performed in the Git repository. It does not matter in which ABAP system you create the branch. However, you have to check it out in each ABAP system where you want to use it, be it for development, testing or productive use.
- In an ABAP system
  - there is only one branch of a software component at a time
  - all developers work on the same object versions
  - an object can be edited by one developer at once
- Pulling a software component does not simply copy object files, but, depending on the object type, also activates the objects, for example a data element, or even generates additional invisible objects in Eclipse, for example authorization artifacts for business catalogs used in business roles.
- You have no merge functionality. If you pull the current branch while changes have been added from another ABAP system, all the changes in the ABAP system the branch is pulled into are discarded. Consequently, you should not work on the same branch in two different ABAP systems.
- Before being able to pull a software component again or before checking out a different branch, you have to release all open transport requests. There is no way of saving them without releasing, like a git stash.
- Releasing a transport request combines the git actions of committing and pushing. The Transport Request Id and description are displayed in the commit message.
- A new branch is created remotely in the Git repository. This means that if the new branch shall be derived from the one currently checked out, changes in the ABAP system that are not yet committed and pushed by releasing the transport that contains these changes are not part of the new branch. The *Last Commit Message* text consists of the Transport Request ID and description, and helps you to ensure that the expected changes are part of the new branch. If you don't want to hold back any new changes, you must release all open transports before creating a new branch.
- If you need to discard changes, you have to do this manually. This can be done with the help of the *History* view in Eclipse, which offers a compare editor to revert to the last transported version. Afterwards, you have to remove all the discarded objects from their modifiable transport request in the *Transport Organizer* view.
- Branches cannot be deleted or set to *obsolete*, therefore consumers have to be informed about a branch that they shall not use, for example, if you skipped a release branch that could not be finished in time.

- Basic change logging using the “traditional” ABAP server capabilities is available in the development ABAP system: e.g. the [Revision History](#) for ABAP objects and the [Responsible](#) and [Last Changed By](#) contacts. However, if you de-provision the development ABAP system, this information is lost. The Git repository still stores the version (see [Information for Audit \[page 763\]](#)) and the person who released the transport request(s) but the information on the [Last Changed By](#) contact per object is only available in the development ABAP system.
- You cannot work offline because you need access to the ABAP system.

#### 4.2.6.4 Working with abapGit

abapGit is an open source developed Git client for the ABAP server to import and export ABAP objects between ABAP systems.

With abapGit, you can export your ABAP objects from any system to another, typically from on-premise to cloud or from one cloud system to another. See [Use abapGit to Transfer Your On-Premise ABAP Source Code to the Cloud](#) and [Push Your ABAP Source Code from SAP BTP, ABAP Environment to a GitHub Repository Using abapGit](#).

##### Note

While the ABAP environment comes with a preinstalled cloud version of abapGit, you can only use abapGit in an on-premise system if you have installed the [official community version](#) of abapGit. However, the official community version is not supported by SAP.

With the official SAP distribution of abapGit, you can use ABAP Development Tools (ADT) in combination with the abapGit feature for ADT, which allows easier access to the import/export functionality provided by abapGit. For more information on how to install and work with ABAP Development Tools, see [Installing ABAP Development Tools](#) and [abapGit Repositories ADT Plug-In](#).

### Use Cases

#### *Side-by-Side-Development*

abapGit can assist you in side-by-side development by enabling the transfer of source code between on-premise systems and SAP BTP, ABAP Environment.

#### *Account Transfer*

abapGit can support you in transferring your development code assets from one Steampunk system to another, particularly across different BTP global accounts.

#### *Partner Code Transfer*

As a partner, you can utilize abapGit to export your development code assets from your development landscape and import them into your customers' development system.

#### *Demo Samples and Open Source Offerings*

You can use abapGit to distribute code as open source, e.g., Code Pal for ABAP - Cloud Edition or demo samples, such as the ABAP Flight Reference Scenario for the ABAP RESTful Application Programming Model.

## Restrictions and Errors

There are certain restrictions to the export and import functionality when using abapGit in the ABAP environment. This is due to the cloud-optimized set of supported ABAP object types that change with each release as new ABAP object types are added continuously. See [Released ABAP Object Types \[page 736\]](#). Depending on your application, you may face the following issues when using abapGit in cloud systems:

- Export errors when pushing unsupported objects
- Import errors when pulling unsupported objects
- Activation errors when activating imported objects

### Export Errors

If you try to export objects that are not supported, you receive a warning during the push operation, and the unreleased object is skipped.

The only work-around for this is to manually copy the missing ABAP objects to the target system. First, you have to create these objects using the same name in the target system. Afterwards, you either have to manually copy the code or reconfigure the objects in the target system.

Some objects are deliberately not exported because they are considered as compiled or generated so that they can be regenerated when activating their originating objects in the target system. For example, certain service definitions (SRVD) are generated from service consumption models (SRVC) so that they are not exported to git.

### Import Errors

When importing objects with abapGit for ADT, you may get error messages in the import log. Possible reasons can be:

- You are importing object types (for example from an on-premise system) that are not supported in the ABAP environment and the programming model of the ABAP environment. You may have to change your application in the cloud environment to work without these objects.
- You are importing objects with missing dependencies. These dependent objects can't be exported because they are not on the list of Released ABAP Object Types. You must copy/recreate these missing dependent objects and restart the import.
- You are importing objects using ABAP namespaces. Make sure these namespaces have been imported into the target system before importing your application with abapGit.
- You are importing a software component relations object. Make sure that a software component with the same name has been created in the target system already. Please note that this object can be imported into the top-level structure package of the corresponding software component only.

#### ⓘ Note

Local changes to ABAP objects that have not been saved are overwritten by the import. Open transport requests for any of the imported ABAP objects must be released before they can be changed. Otherwise, they get locked.

### Activation Errors

All objects imported with abapGit are imported in an inactive state, except the ones that do not offer an inactive state, to ensure that their consistency can be properly checked during activation. Hence, you have to activate

all imported objects after importing them so that ABAP activation can verify that only released APIs have been used.

During the activation process, some objects may depend on other objects, which need to be activated beforehand. However, not all object types consider the objects they depend on during mass activation. For example:

- Data dictionary (DDic) and core data service (CDS) objects do not always resolve their dependencies in the first mass activation attempt so that you have to activate them in multiple cycles.
- Service bindings need to be activated after service definitions.

If activation issues occur, repeatedly try to activate objects in the right sequence. If that does not work, try reimporting and reactivating objects multiple times until all objects of your application are active.

#### → Tip

If you face any issues with the official abapGit distribution of SAP, create an incident using component BC-CP-ABA.

For issues related to the community open-source distribution of abapGit, create an issue at <https://github.com/abapGit/abapGit/issues>.

#### ⓘ Note

It is possible to define the external systems, the ABAP system is allowed to communicate with, by maintaining a list of trusted certificate authorities. Please use the Maintain Certificate Trust List app to do so. This app can be used by administrators, users with the SAP\_BR\_ADMINISTRATOR role. For further information, please go to [Maintain Certificate Trust List \[page 2771\]](#)

## Working with UI Artifacts

As described in [Restrictions and Errors](#), the originating objects of the UI/SAP Fiori artifacts (SMIM, UIAD, WAPA) have their own lifecycle in a separate Git repository. The deployment to the target ABAP system will be done via SAP Business Application Studio / Visual Studio Code using SAP Fiori Tools.

For the case that one of the above-described use cases applies and you must transfer your application, remember that the UI artifacts have their own Git repository. This is important because abapGit does not take the UI objects into account.

Nevertheless, you have two different Git repositories. The Git repository with ABAP artefacts and the Git repository with the UI artefacts.

### Important to Know

Understanding the unique usage of the UI Git repository compared to the abapGit repository is crucial. The main difference lies in the fact that the user interface (UI) maintains its own lifecycle. This means any modifications or updates to the UI must be made within the Business Application Studio or Visual Studio Code and then pushed to the dedicated UI repository.

The abapGit repository is usually only used once to (additionally) save and share your code. The lifecycle happens in your ABAP system.

## Procedure

Depending on your needs, you might have to duplicate the repositories, create branches, or fork branches. For example, to transfer an application from a partner development system to a customer development system, you can use the export and import functionality of your Git provider. Alternatively, you can utilize Git functionality like branching or forking to "duplicate" the repository.

For additional information on how to duplicate the Git repositories, it may be helpful to consult the documentation provided by your Git provider.

### *Import Steps*

- You have read access to both Git repositories (abapGit and UI)
- Link and pull the abapGit repository into your ABAP system.
- Click onto the button [Activate All](#) to activate all objects.

#### ⓘ Note

The activation might fail due to missing dependencies, e.g., missing UI artifacts.

- Clone your UI repositories to Business Application StudioAS/Visual Studio and deploy the UI to your ABAP system.
- Pull the abapGit repository again because some abapGit objects rely on the UI objects.
- Click onto the button [Activate All](#) to activate all objects.

#### ⓘ Note

Please be aware that it might be necessary to repeat the last two steps.

## Related Information

[Released ABAP Object Types \[page 736\]](#)

### 4.2.6.4.1 Install and Set Up abapGit

Learn how to install and set up abapGit.

- You have signed up for a Git account of your choice, for example GitHub.
- You have downloaded and installed the front-end components of [ABAP Development Tools \(ADT\)](#).

#### ⚠ Restriction

Please be aware that abapGit is an open-source project maintained by the open source community and is not owned by SAP. Therefore, SAP does not provide support for the open source abapGit version (ZABAPGIT). Consequently, we ask you to refrain from creating incidents related to the use of the ZABAPGIT report. However, for the SAP BTP, ABAP environment, a separate SAP-owned version (fork) of abapGit has been established. SAP does provide standard support for this version.

#### *Part 1: Create a Git repository*

1. Log on to your github.com account.
2. Create an abapGit repository by clicking on *New repository*.
3. Enter a repository name, tick the *Initialize this repository with a README* checkbox, and select *Create repository*.

#### *Part 2: Install zabapGit for the on-premise use case*

1. Copy the content of the latest build from program zabapgit to your clipboard. You can find the content in the abapGit repository <https://github.com/abapGit/abapGit>.

##### Caution

Check with your system administrator before installing zabapgit.

2. Log on to an on-premise system of your choice, create a new program, and paste the content from your clipboard.

##### Note

Select EN as your logon language.

3. Activate and execute the program.

abapGit is installed and launched.

#### *Part 3: Install an ADT plugin for abapGit*

1. Log on to ABAP Development Tools in Eclipse.
2. Navigate to  .
3. To install the abapGit repositories ADT plug-in, add the following URL: <https://eclipse.abapgit.org/updateSite/> and provide a name for the repository.
4. To display all the available features, press enter, and select *abapGit for ABAP Development Tools (ADT)*.
5. Select *Next* to finish the installation.

## Related Information

[Create Content in an On-Premise System and Push it to abapGit Repository](#)

### 4.2.6.4.2 Create Content in an On-Premise System and Push it to abapGit Repository

## Prerequisites

You have installed and set up abapGit. See [Install and Set Up abapGit \[page 723\]](#).

### **⚠ Restriction**

Please be aware that abapGit is an open-source project owned by the community. Therefore, we do not provide support for abapGIT. We only support the Git integration in the ABAP environment.

## **Procedure**

1. After installing and launching abapGit, select [Clone or download](#), and copy the URL of your repository.
2. Call up transaction ZABAPGIT, and select [+ Online](#).
3. Paste the URL of your repository.
4. Select [Create package](#).

### **ⓘ Note**

If you have already created a software component with gCTS, you can skip steps 4 to 6 and use this software component in step 7.

5. Add a package name and short description, and select [Continue](#).
6. Confirm with [OK](#).  
The cloned abapGit repository is displayed in abapGit.
7. Log on to ABAP Development Tools in Eclipse, and navigate to your newly created package.
8. Add ABAP development objects to your package. See [Released ABAP Object Types \[page 736\]](#).
9. Navigate back to the abapGit UI, and select [Refresh](#) to display the development objects that you have created in Eclipse.
10. Choose [Stage](#).
11. Select single objects or choose [Add all and commit](#).

### **ⓘ Note**

You can also select structure packages to be exported.

12. Enter a [committer name](#), [committer e-mail](#), and [comment](#).
13. Select [Commit](#).
14. In the Login popup, enter your abapGit repository server credentials, and select [Execute](#).

The pushed ABAP objects are displayed in your abapGit repository.

## **Next Steps**

[Import Content from abapGit Repository into the ABAP Environment \[page 726\]](#)

## 4.2.6.4.3 Import Content from abapGit Repository into the ABAP Environment

Learn how to import content from your abapGit repository into your ABAP environment, and transfer it across multiple instances.

### Prerequisites

- You have installed the abapGit repositories ADT plug-in. See <https://eclipse.abapgit.org/updatesite/>.
- You have created content in your on-premise system and pushed it to your abapGit repository. See [Create Content in an On-Premise System and Push it to abapGit Repository \[page 724\]](#).

#### ⓘ Note

The repository has to be accessible from the Internet. Cloud Connector is not supported.

- You have access to an ABAP cloud system. See [Creating an ABAP System \[page 180\]](#).
- You have assigned a developer user in the ABAP environment to the developer role. See [Assigning the ABAP Developer User to the ABAP Developer Role \[page 192\]](#).

#### ⚠ Restriction

Please note that abapGit is an open-source project owned by the community. Therefore, we do not provide support for abapGit. We only support the abapGit integration in the ABAP environment.

### Procedure

1. Log on to ABAP Development Tools in Eclipse.
2. In the *Project Explorer*, select your cloud project system, and navigate to to open the abapGit repositories view.
3. Search for *ABAP*, choose *abapGit Repositories*, and select *Open*.
4. In the *abapGit repositories* view, select the clone button (green + icon).
5. Enter your abapGit repository URL, and select *Next*.
6. Select a *Branch* and *Package*, where you want your abapGit repository to be cloned, and confirm with *Next*.

#### ⓘ Note

If there are no packages, you have to create a structure package and add a development package.

7. Select the default transport request, and choose *Finish*.  
The imported objects are displayed in your package. See [Released ABAP Object Types \[page 736\]](#).

#### ⓘ Note

The number of imported objects can differ from the number of exported objects because only released ABAP object types are considered during the import.

## 4.2.6.4.4 Folder Logic in abapGit

### Folder Logic

AbapGit in Steampunk offers two folder logics, either “Prefix” or “Full”. The folder logic has to be selected when linking a repository in ABAP Developer Tools in the popup. Afterward, the information can also be found in the abapGit Repositories ADT plugin. The folder logic information defines the way how subpackages of the root package that needs to be linked will be converted to folders in the linked repository. Therefore, this setting affects only the folder structure in the remote repository and has no implication to the object structure in the ABAP system.

#### Prefix

The folder logic PREFIX allows the installation of a repository into a different parent package (in different systems). This can even be a local package (ZLOCAL), in which case no transport request is required. A package name must contain its parent package name as a prefix. This means that the names of the subpackages will start with the name of the parent package, for example:

- TESTMYTEST01 (parent)
- TESTMYTEST01\_RUN\_01 (child1)
- TESTMYTEST01\_RUN\_02 (child2)

#### ⓘ Note

The prefix folder logic will not work correctly if you do not follow the convention.

Other examples, including valid package prefix:

- ZFOO
- ZFOO\_BAR
- ZFOO\_BAR\_QUX

This will produce the following folder structure:

```
/src  
/src/bar  
/src/bar/qux
```

Invalid package prefix:

- ZFOO
- ZBAR

## Full

The folder logic FULL forces the installation of a repository into packages with exactly the same name. Any package name is accepted. Folders will be named like packages. Any package name is accepted

- ZBASE
- ZSOMETHING
- ZHELLO

This will produce the following folder structure:

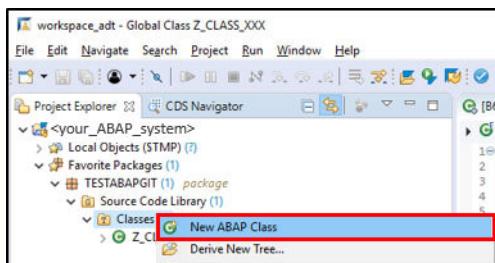
```
/src  
/src/zsomething  
/src/zsomething/zhello
```

In general, the folder logic setting is stored in the *.abapgit.xml file* in the remote repository, too. It is highly recommended to take over this information during the linkage process, in case the file already exists. Otherwise, a mixture of different folder logic information on the local and remote side would lead to inconsistencies.

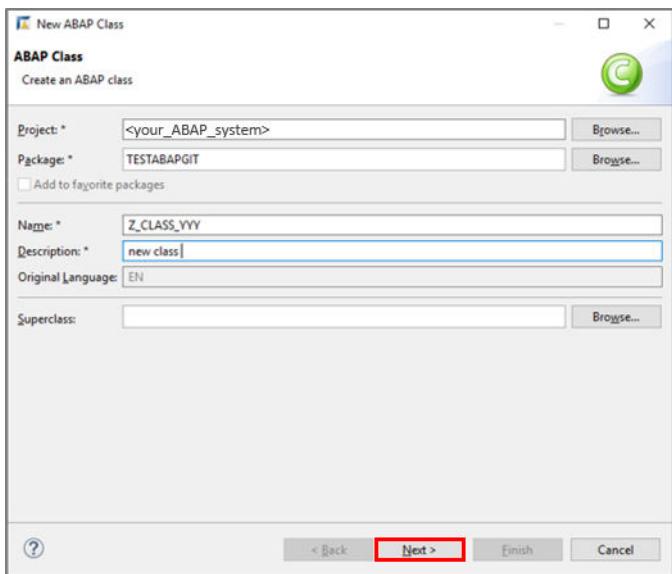
## 4.2.6.4.5 Transfer your ABAP Source Code via abapGit

Use abapGit to transfer your ABAP source code from an ABAP environment instance back to your repository.

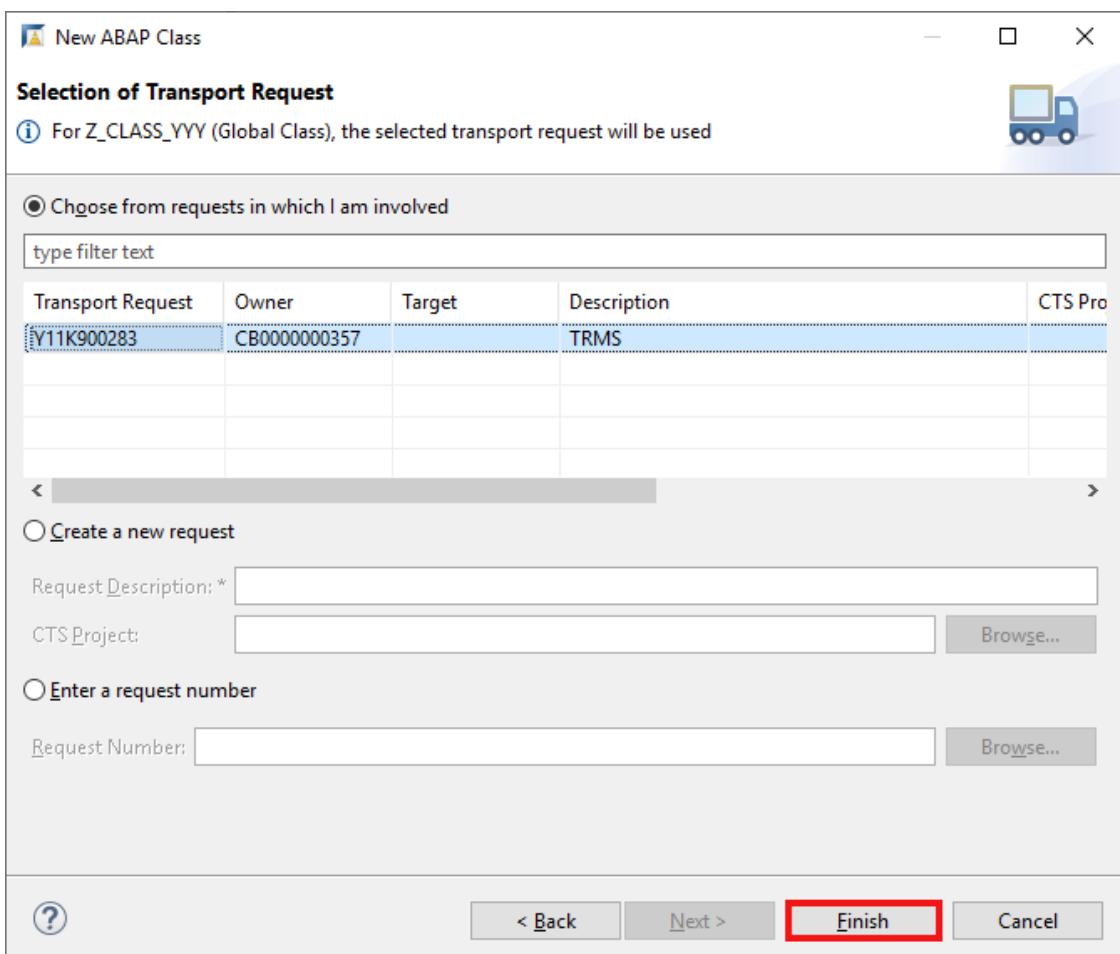
1. Create or adapt your source code in the ABAP environment instance:
  1. Open **ABAP Development Tools** and logon to your ABAP system.
  2. Add an ABAP development object, e.g. an ABAP class, to your already existing **TESTABAPGIT** package.  
To do this, click on **Classes** and select **New ABAP Class**.



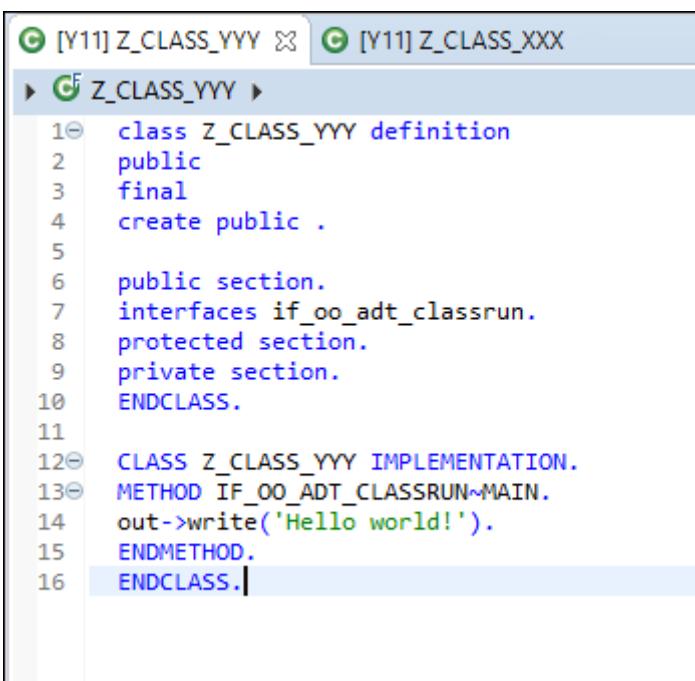
3. Enter a name and description for your new ABAP class and click **Next**.



4. Click *Finish*.



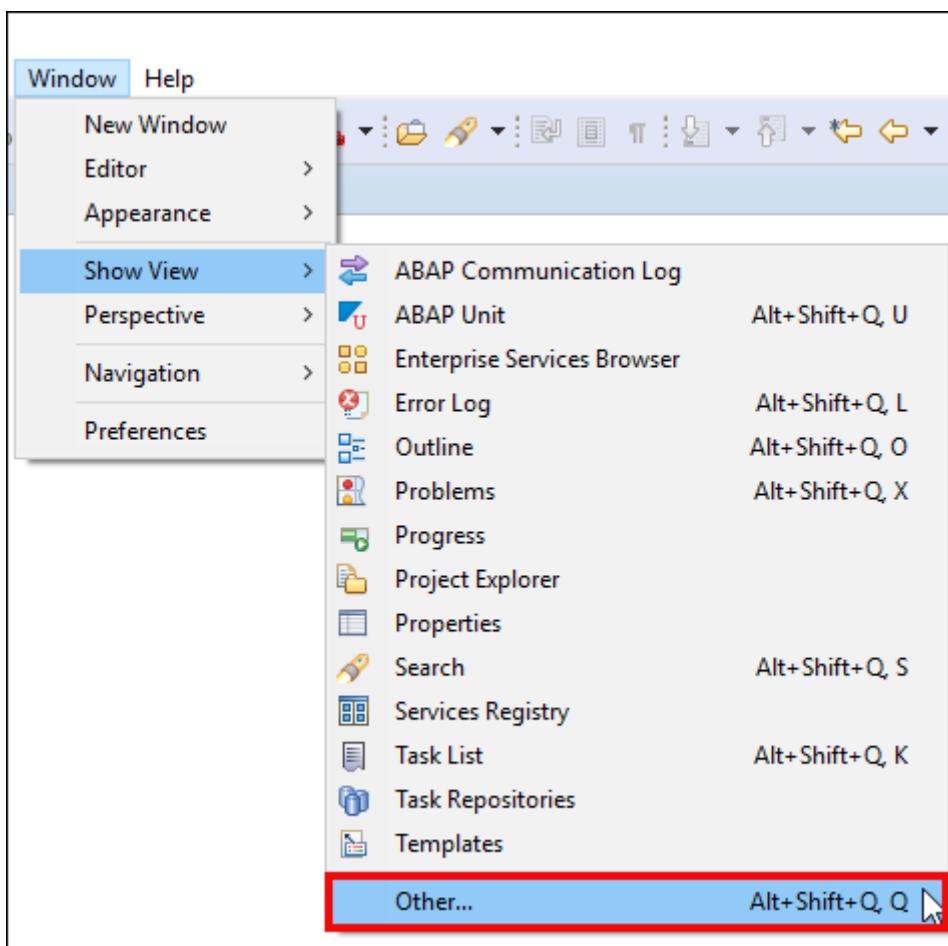
5. Now you can implement your newly created class.



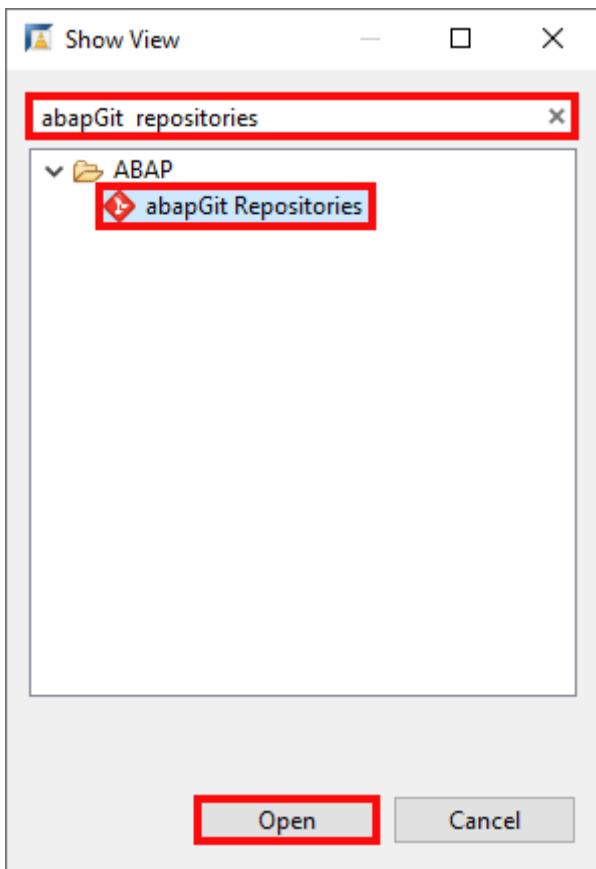
The screenshot shows the SAP ABAP IDE interface. At the top, there are two tabs: [Y11] Z\_CLASS\_YYY and [Y11] Z\_CLASS\_XXX. Below the tabs, the code editor displays the following ABAP code:

```
1①  class Z_CLASS_YYY definition
2    public
3      final
4      create public .
5
6      public section.
7      interfaces if_oo_adt_classrun.
8      protected section.
9      private section.
10     ENDCLASS.
11
12②  CLASS Z_CLASS_YYY IMPLEMENTATION.
13③  METHOD IF_OO_ADT_CLASSRUN~MAIN.
14    out->write('Hello world!').
15  ENDMETHOD.
16  ENDCLASS.
```

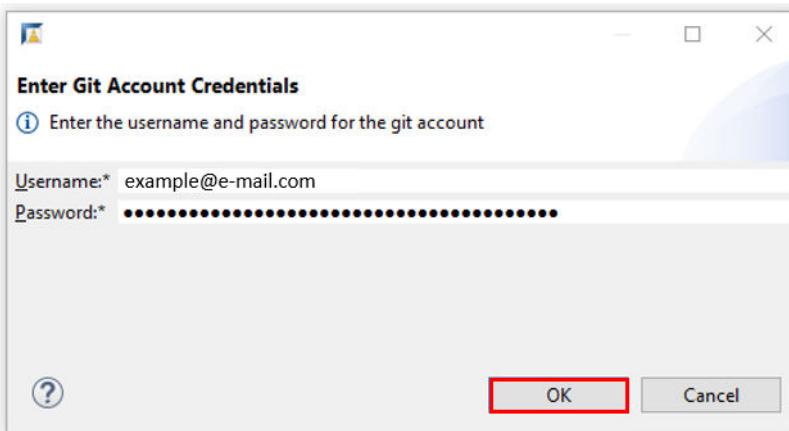
6. Save and activate.
2. Open abapGit repositories:
  1. Select your **ABAP system** in the Project Explorer and select **Windows > Show View > Other...** to open the *abapGit* repositories.



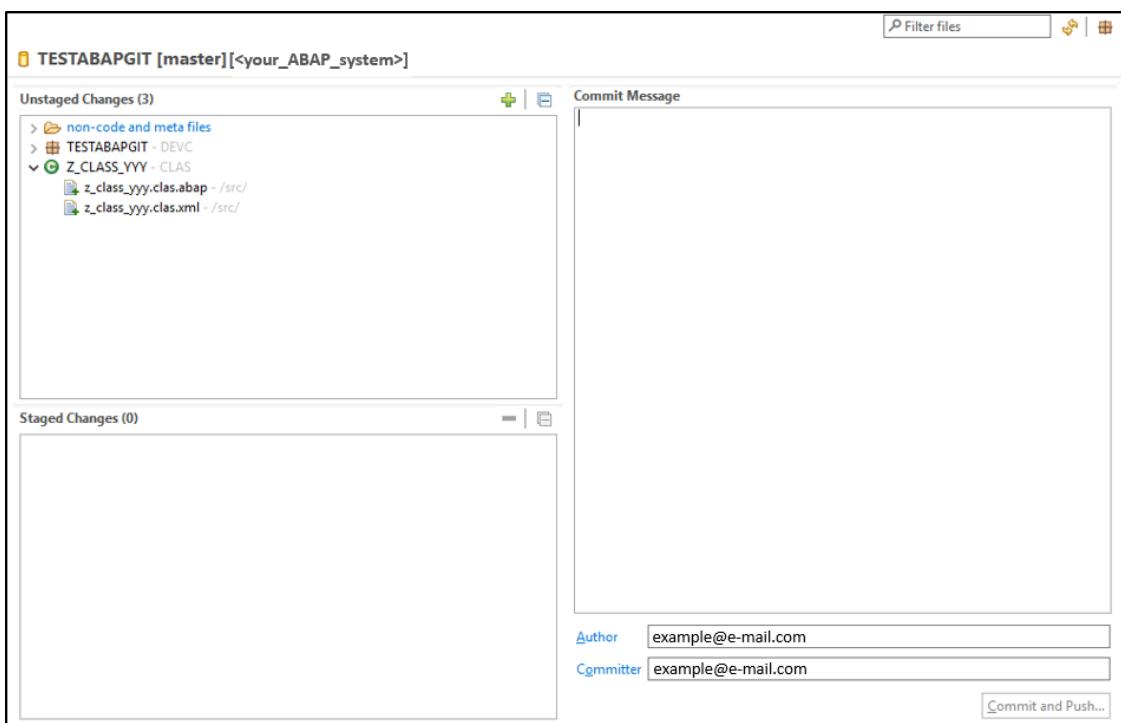
2. Search for *abapGit repositories*, select it and click *Open*.



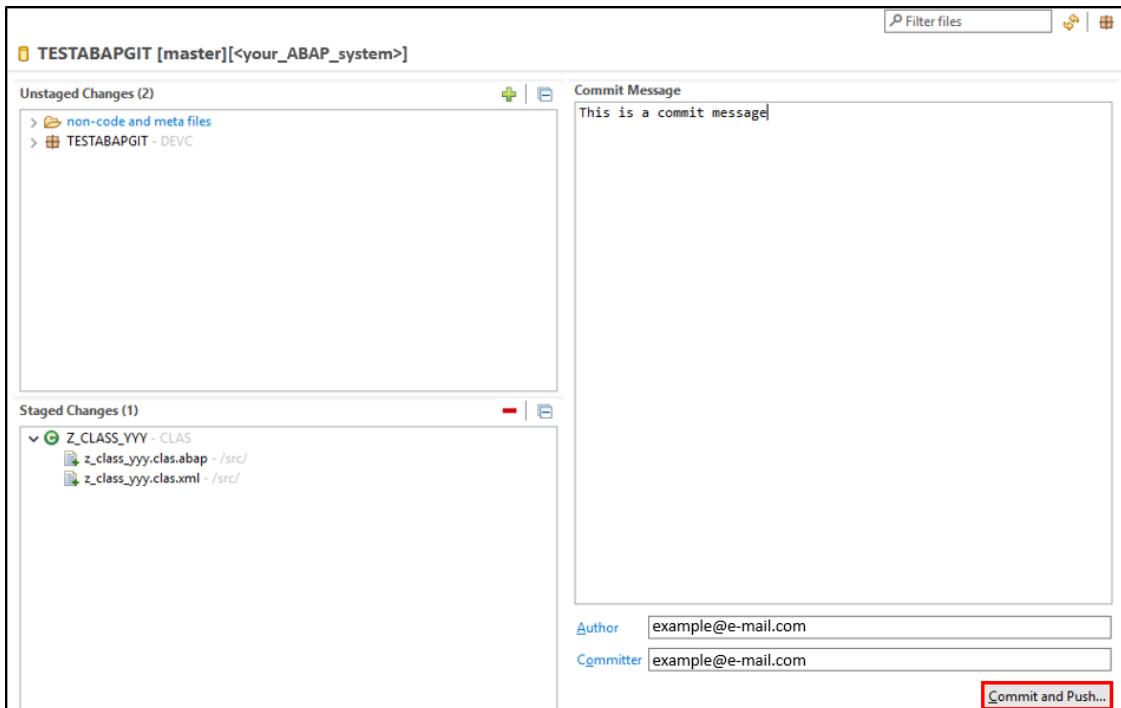
3. Stage and commit your ABAP development objects.
  1. Right-click your *TESTABAPGIT* package and click *Stage and Push*.
  2. Enter your repository credentials in the popup and click *OK*.



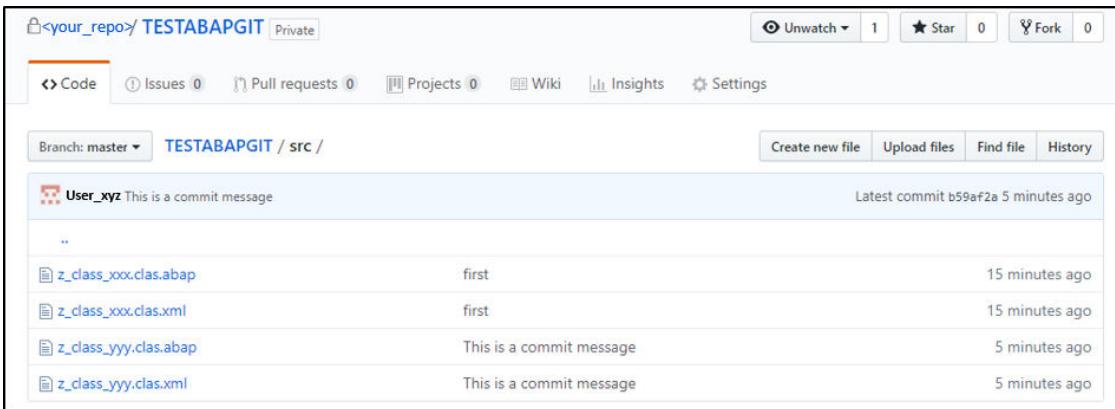
3. In the staging view you will see a list of changed, created or deleted ABAP development objects which can be transferred to your repository. The icon of the respective file indicates whether a file was created, modified, or deleted.



4. ABAP development objects which should become staged can be selected by drag and drop from the *unstaged* box to the *staged* box or by clicking on the **+** button. Enter your commit message and click **Commit and Push**.



5. You'll see a notification telling you that your push has started. Click **OK**.
6. The staged objects have now been transferred to your repository.



#### 4.2.6.4.6 Object Deletions with abapGit

Initially, abapGit as a Git client has been integrated into SAP BTP, ABAP environment to enable customers to facilitate the import of code assets from customers' on-premise systems to SAP BTP, ABAP environment. Over time, additional use cases came up, so that the focus is no longer on a one-time transfer but also on repeated export and import of development objects across different landscapes, including handling of deletions.

In general, there are two basic types of deletions.

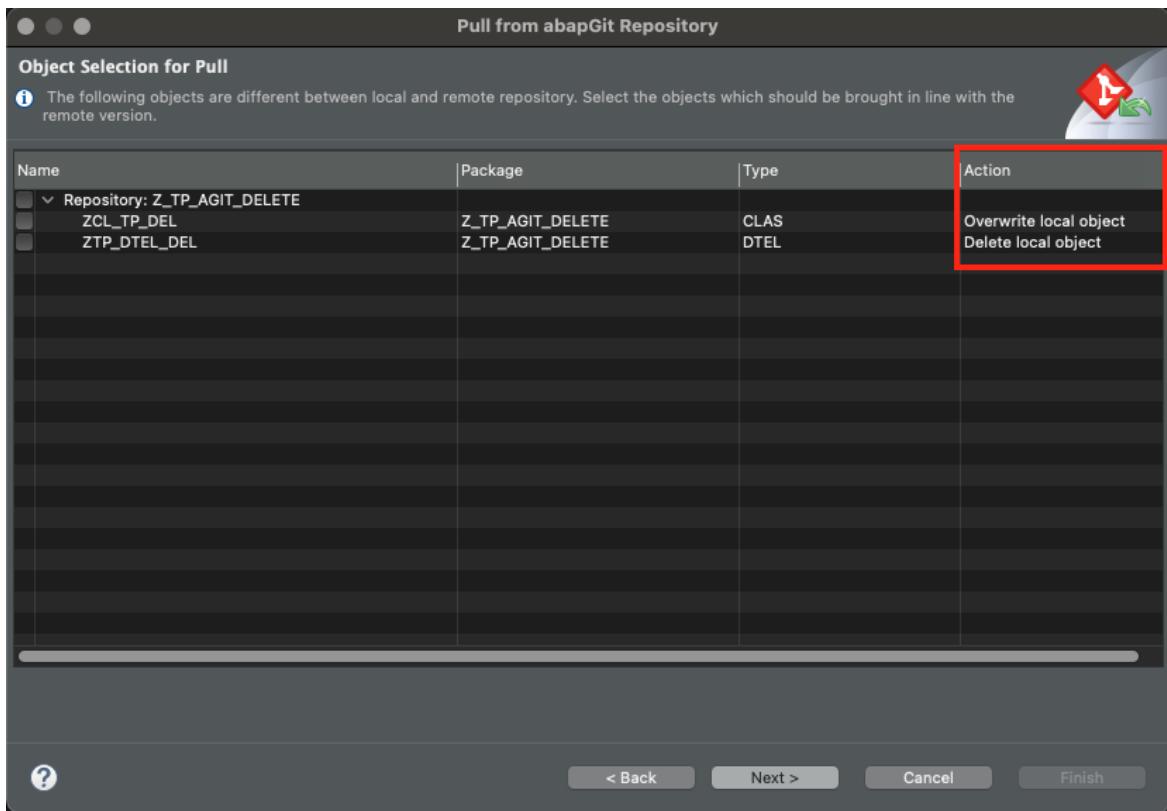
Deletion Type	Git Action	Leading Repository	Determination logic
Local	Pull	Remote	If an object exists locally but not remotely, it will be marked for deletion during the pull process.
Remote	Stage & Push	Local	If an object exists remotely but not locally, it will be marked for deletion during the staging and push process.

#### Local Deletion

##### Prerequisites

Objects from the remote repository have been pulled into your linked ABAP package.

1. Start by deleting the file(s) of your objects in your abapGit remote repository.
2. In ABAP Development Tools for Eclipse, you initiate a pull run.  
After that, the system performs a delta calculation to identify the differences between the package and the remote repository.
3. The object previously deleted in your remote repository will now be shown as the action [delete local object](#). Here, you can select the object to be deleted.  
Afterwards, click on [Next](#) and then [Finish](#).



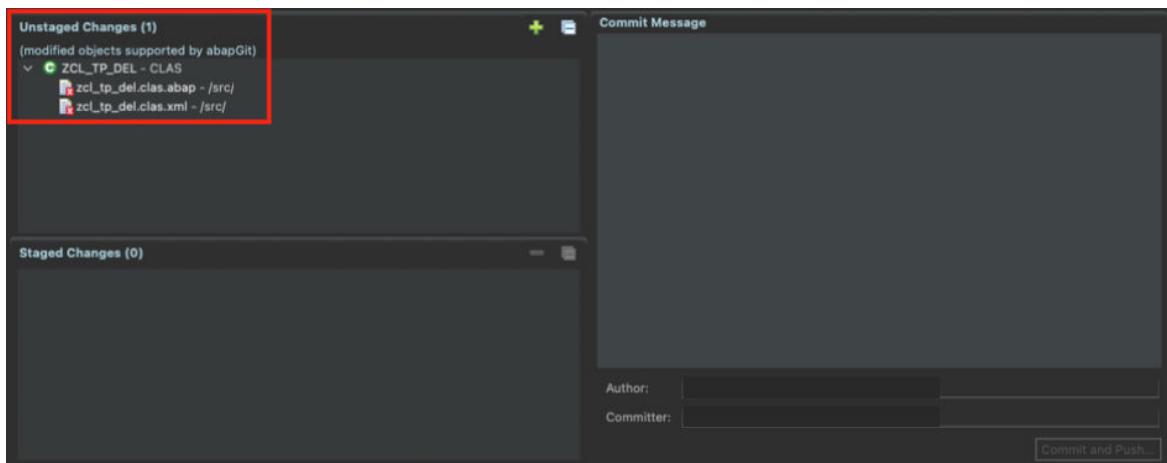
4. After the pull run, you can check the [ABAP Object Log](#) for the successful deletion.
5. Finally, reload the [Project Explorer](#) to see that the object has disappeared in the overview as well.

## Remote Deletion

### Prerequisites

Objects from the remote repository have been pulled into your linked ABAP package.

1. Start by deleting your object locally in ABAP Development Tools for Eclipse first.  
In the [Project Explorer](#), select the according object and delete it. This opens up a window called [Select Objects](#). Here, you select the object to be deleted and then [Finish](#).
2. Next, initiate a stage and pull of the emerging changes.  
To do this, right-click your abapGit package in ABAP Development Tools for Eclipse and select [Stage and Push](#). The following picture shows an example overview of unstaged and staged changes.



- In the changes overview, the previously deleted object in your package will now be shown with a deleted icon.

You can drag and drop the object from *Unstaged Changes* to *Staged Changes*.

Afterwards, select *Commit & Push*.

In the tab *abapGit Repositories*, you can now see that your linked repository will show the Status *Pushed Successfully*.

#### 4.2.6.4.7 Released ABAP Object Types

The following table contains all the released ABAP object types that can be imported into your ABAP environment tenant.

##### Note

If you try to import or export objects that are not supported, you receive a warning during the transport operation. The objects that are not released are skipped and not transported. To avoid this, you have to manually copy the ABAP objects to the target system. Note that you have to create these objects using the same name in the target system, unless they already exist. Afterwards, you either have to manually copy the code or reconfigure the objects identically in the target system.

APIS	API Release State of Objects
APLO	Application Log Object
AUTH	Authorization Check Fields
BDEF	Behavior Definition
BGQC	Background Processing Context
CDBO	Customer Data Browser Object
CFDF	Custom Field
CHDO	Change Document
CHKC	ATC Check Category

CHKO	ATC Check Object
CHKV	ATC Check Variant
CLAS	Class
COTA	Connection Target Transport Object
DCLS	ABAP Data Control Language Sources
DDLS	Data Definition Language Source
DDLX	Core Data Services Metadata Extensions
DEVc	Package
DOMA	Domain
DRTY	Dictionary: CDS Type Definitions
DTEB	Dictionary Tuning: Entities Buffer
DTEL	Data Element
EEEC	Event Consumption Modell
ENHO	Enhancement Implementation Object
ENHS	Enhancement Spot Object
EVTB	Event Binding
ENQU	Lock Object
FUGR	Function Group
FUNC	Function Module
GSMP	Metric Provider
HTTP	HTTP Service
INTF	Interface
MSAG	Message Class
NONT	SAP Object Node Type
NROB	Number Range Object
RONT	SAP Object Type
SAJC	Application Job Catalog Entry
SAJT	Application Job Template
SCO1	Communication: Communication Scenario
SCO2	Communication: Inbound Service
SCO3	Outbound Service
SIA1	IAM: Business Catalog
SIA2	IAM: Restriction Type
SIA3	IAM: Authorization Object Extension
SIA5	IAM: Restriction Field
SIA6	IAM: App
SIA7	IAM: Business Catalog App Assignment

SIA8	IAM: Business Role Template
SIA9	IAM: Business Role Templ. Bus. Catalog Assignment
SIAD	Business Role Template Fiori Space Assignment
SKTD	Knowledge Transfer Document
SMBC	Maintainable Business Configuration
SOD1	API Package
SOD2	API Package Assignment
SRVD	Service Definition
SUSH	Assignment: Service --> Authorization Objects
SUSO	Authorization Object
SWCR	Software Component Relations
TABL	Table Definition
TTYP	Table Type
UIAD	Application Descriptors for the Fiori Launchpad Environment
UIPG	Fiori Launchpad Page Template
UIST	Fiori Launchpad Space Template
XSLT	Transformation

## Restrictions

- All objects imported with abapGit are imported as inactive, except the ones that don't have an inactive state. Therefore, you have to activate all the imported objects after the import.

### ⓘ Note

During the activation process, some objects may depend on other objects which need to be activated beforehand. For instance, the newly enabled DSFI objects depend on a corresponding DSFD, that needs to be activated beforehand.

Also note that mass activation is not supported.

- All local changes to ABAP objects that are not saved are overwritten by the import.
- Open transport requests for any of the imported ABAP objects must be released before they can be changed. Otherwise, they get locked.

## Known Issues

- Changes in table definitions could lead to data loss.

## 4.2.6.5 Setting Up and Working with Your Landscape

Depending on your requirements, you need to adjust the setup of your software lifecycle management and the underlying landscape. The following sections give you examples for the most common setups.

In general, the number of ABAP systems in your landscape is the result of the processes (development (DEV), quality assurance (QAS), and production (PRD)) that need to run in parallel and the number of branches of your biggest software component that is worked on or tested in parallel.

The number of ABAP systems = (number of branches x DEV systems) + (number of branches x QAS systems) + (number of PRD systems).

This is due to the fact that an ABAP system always checks out only one branch of a software component. Yet, several software components represented by one branch can run in one ABAP system in parallel.

### 4.2.6.5.1 Required Tools

Different actions require different tools. In the process descriptions, the following tools are used:

- SAP BTP cockpit (web-based administration interface)
- SAP Fiori app [Maintain Business Roles \[page 2666\]](#)
- SAP Fiori app [Maintain Business Users \[page 2655\]](#)
- SAP Fiori app [Manage Software Components \[page 2805\]](#)
- SAP Fiori app [Export Customizing Transports \[page 2582\]](#)
- SAP Fiori app [Maintain Business Configurations \[page 2574\]](#)
- Eclipse Tool for the ABAP environment is an Eclipse installation with ABAP Development Tools (ADT) plugin (see [Eclipse Tool for the ABAP Environment](#))
- Custom SAP Fiori apps: these apps are the result/aim of your development
- External test tools, for example for OData tests
- External documentation tool to document test results and release decision if required

### 4.2.6.5.2 Required Business Roles

Different actions require different roles as well. In the process description, the following role proposals are used:

- System Administrator
  - Creates ABAP systems in the SAP BTP cockpit
  - Needs authorization for space/organization management in the Cloud Foundry environment subaccount
- User Administrator
  - Maintains business roles and assigns users to them
  - Needs authorization for business catalogs

Catalog ID	Catalog Description	Needed For
SAP_CORE_BC_IAM_UM	Identity and Access Management – User Management	Creating and maintaining business users with <i>Maintain Business Users</i> app
SAP_CORE_BC_IAM_RM	Identity and Access Management – Role Management	Maintaining business roles with <i>Maintain Business Roles</i> app

- Release Manager
  - Performs tasks that are executed centrally for a release, such as software component creation and import, making the release decision, approving corrections, creating and releasing transports etc.
  - Needs authorization for business catalogs

Catalog ID	Catalog Description	Needed For
SAP_A4C_BC_MSCL_LIFECYC	Creating Software Components	Creating software components with <i>Manage Software Components</i> app
SAP_A4C_BC_TRN_REL_PC	Developing Transport Requests in ABAP	Development Tools for Eclipse
SAP_A4C_BC_DEV_BCT_REL_P	Development Objects	Development objects in transport organizer in ABAP Development Tool
SAP_CORE_BC_IAM_UM	Identity and Access Management - User Management	Un-/locking developers for corrections in correction ABAP systems with <i>Maintain Business Users</i> app
SAP_CORE_BC_IAM_RM	Identity and Access Management – Role Assignment	En-/disabling developers for corrections in correction ABAP systems by role assignment with <i>Maintain Business Users</i> app
SAP_CORE_BC_BCT_TRN_REL_B	Business Configuration Transport Release Management	Creating, editing, transporting and releasing customizing transports with <i>Export Customizing Transports</i> app

- Developer
  - Develops the code and performs tests
  - Needs authorization for business catalogs

Catalog ID	Catalog Description	Needed For
SAP_A4C_BC_DEV_PC	Development – ABAP Development Tools	Developing in ADT for Eclipse
SAP_A4C_BC_TRN_MNG_PC	Development – Transport Management	Creating transport requests in ADT for Eclipse

- Tester
  - Tests the software

### Note

Depending on your organization, the tasks of this role can be performed by automatic test tools, developers, or dedicated persons that are responsible for executing manual tests.

- Needs authorizations for custom business catalogs created via IAM business catalogs for custom apps.

### Caution

Do not assign catalog SAP\_CORE\_BC\_EXT\_TST (Extensibility - Custom Apps and Services) to a tester.

As this catalog is used to provide an easy test of custom apps or their underlying services in a development ABAP system only by adding them automatically to it during activation, real service tests with own catalogs might not be tested securely. In non-development systems, automatic authorization via business catalog SAP\_CORE\_BC\_EXT\_TST is not enabled, instead custom authorization is required.

- Business Configuration Expert
  - Maintains and tests business configuration

Catalog ID	Catalog Description	Needed For
SAP_CA_BC_IC_LND_PC	Customizing - Business Configuration	General business configuration tasks, such as uploading business configuration or viewing the change log for business configuration
SAP_CORE_BC_BCT_TRN_MNG_PC	Business Configuration - Transport Management	Creating and editing customizing transports
Application-specific catalog with configuration apps, for example SAP_CA_BC_IC_LND_NUM_PC	Example: Number Range Management - Configuration	Configuring number ranges

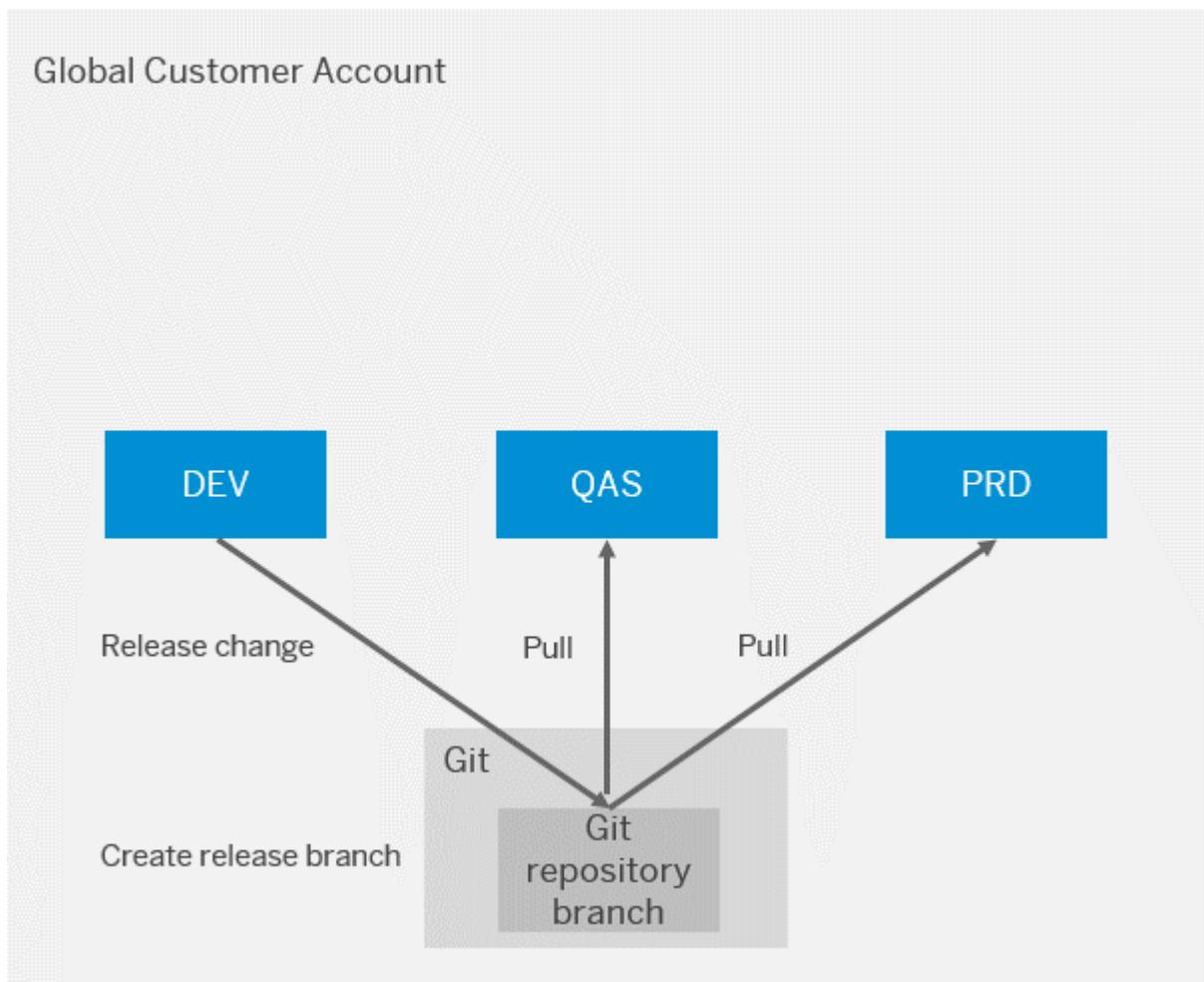
## 4.2.6.5.3 Important Notes

- The creation of a Git repository branch creates a clone of the parent branch. If you create a branch immediately after the release decision, you have a “frozen” state. You can import this state safely into one or multiple production systems, “pre-production” systems, etc., or you may use it later for auditing.
- In the different system setups, the production systems can either be provisioned once development starts or later on, when needed for the first time for the Go Live, which is the point in time, when the first release is used productively in the production system.
- A classical 3-system setup comprising a development, quality assurance, and production system can be enhanced by pre-production systems, multiple test systems, etc. Test systems can be provisioned on demand (temporary) or permanently.

#### 4.2.6.5.4 Use Case 1: One Codeline in a 3-System Landscape

You can apply this setup if you have occasional development activities for larger applications where testing needs to run in parallel to development or should take place in a non-development system to ensure the solution also runs in a non-development system. In this setup, you either need to be able to pause development for a fix that has to be delivered before the next release or you have to deliver fixes as part of the next possible release.

This landscape consists of a development, quality assurance, and production system.



#### For Go Live/Development After Go Live (Including Deferrable Corrections)

##### Starting situation for Go Live

The Go Live process is characterized by creating different systems only when needed for the first time, however, you can provision the systems already beforehand. Furthermore, the software component of a

planned solution does not exist from the beginning. The resulting release branch is YYYY-01. Apart from this, the Go Live process does not differ from the release development processes afterwards.

### Starting situation after Go Live

- Development system DEV is based on the main branch
- Quality Assurance system QAS and production system PRD are based on the latest release branch YYYY-<nn>. In case of a first release after the Go Live, YYYY-<nn> is YYYY-01

System QAS has always the same software state as the PRD system, unless a new change is tested and released. This means, transport requests are released in development ABAP systems only if development is completed and it is planned to import the changes to the production system.

This process can also be used for deferrable corrections, which do not need to reach production before the next development release. These corrections are handled like regular development.

Step	System	Role	Task	Tool
0	DEV	Release Manager	<p>At Go Live only: Create the software component and clone it initially</p> <p>If required, create a customizing transport request and tasks for the relevant business configuration expert(s).</p>	<a href="#">Manage Software Components app</a> <a href="#">Export Customizing Transports app</a>
1a	DEV	Developer	Develop new functionality or a deferrable correction. All changes are collected in workbench transport requests	ABAP Development Tools for Eclipse
1b	DEV	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests	<a href="#">Maintain Business Configurations app</a>
2 a	DEV	Developer in ADT	Release the development transport request(s).	ABAP Development Tools for Eclipse: Transport Organizer
2 b	DEV	Release Manager	Release only export customizing transports.	<a href="#">Export Customizing Transports app</a>
2 c	DEV	Release Manager	The changes are now in the main branch	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>

Step	System	Role	Task	Tool
2 d	DEV	Release Manager	The released API has to be part of development and the development has to be finished.  Snapshots are pulled after development after the API objects are released. The snapshot has to be set to check-relevant. Then, the ATC compatibility check should run.	<a href="#">Manage API Snapshots app</a>
3	QAS	Release Manager	At Go Live: Pull software component(s) into system QAS.  After Go Live: Check out main branch of software component(s)	<a href="#">Manage Software Components app</a>
4	QAS	Tester	Test the change and report test result  If changes are required, repeat steps 1-4	ABAP Development Tools for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
5	QAS	Release Manager	Release decision: the changes are successfully tested and approved	External documentation tool
6	QAS	Release Manager	Create a release branch YYYY-<nn+1> for each software component (at Go Live: YYYY-01)	<a href="#">Manage Software Components app</a>
7	QAS	Release Manager	Check out the new release branch YYYY-	<a href="#">Manage Software Components app</a>

Step	System	Role	Task	Tool
8	PRD	Release Manager	Check out the new release branch YYYY-<nn+1> for each software component (at Go Live: YYYY-01) into system PRD	<a href="#">Manage Software Components app</a>

## Urgent Corrections

- Development system DEV is based on the main branch
- Quality assurance system QAS and production system PRD are based on the latest release branch YYYY-<nn>. In case of a correction after the Go Live before the second release, YYYY-<nn> is YYYY-01

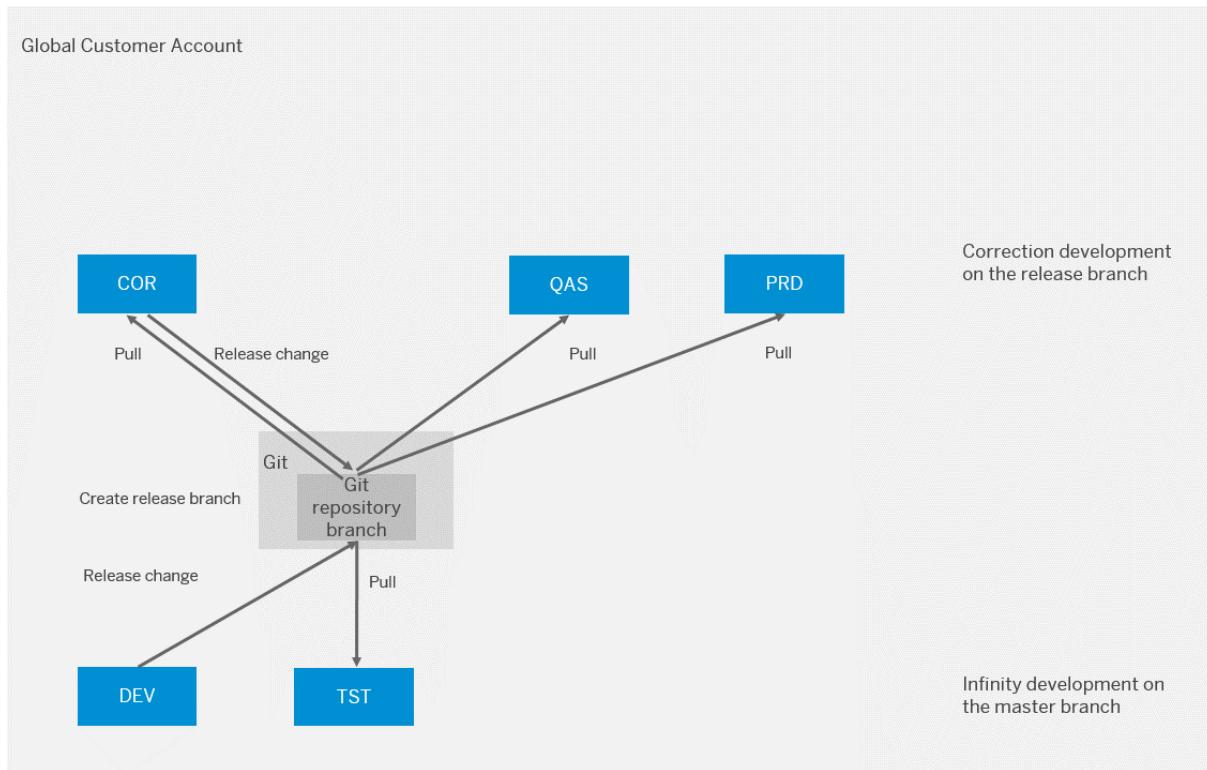
This process differs from the previous one in the branch it is developed in. As the correction is too urgent to release it with the next development release only, it is done in the release branch. To achieve this separation, all current development activities need to be paused because the DEV system needs to check out the latest release branch instead of the main branch. That means all open transports in DEV need to be released first to save the work in progress to the main branch, so that feature development can be resumed later after the correction.

Step	System	Role	Task	Tool
1	DEV	Release Manager	Check out the release branch YYYY-<nn> for each software component  If required, create a customizing transport request and tasks for the relevant business configuration expert(s)	<a href="#">Manage Software Components app</a>  <a href="#">Export Customizing Transports app</a>
2a	DEV	Developer	Fix existing functionality. All changes are collected in workbench transport requests	ABAP Development Tools for Eclipse
2b	DEV	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests	Custom Business Configurations app
3a	DEV	Developer	Releases the development transport requests.	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>

Step	System	Role	Task	Tool
3b	DEV	Release Manager	Releases the customizing transport requests.	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports</a> app
4	QAS	Release Manager	Pull the software component(s) to get the correction into the already checked out release branch YYYY-<nn>	<a href="#">Manage Software Components</a> app
5	QAS	Tester	Test the change and report the test result	ABAP Development Tools for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
			If changes are required, repeat steps 2-5	
6	QAS	Release Manager	Fix is successfully tested and approved	External documentation tool
7	PRD	Release Manager	Pull the software component(s) to get the correction into the already checked out release branch YYYY-<nn>	<a href="#">Manage Software Components</a> app
8	DEV	Release Manager	Check out the main branch in system DEV for each software component	<a href="#">Manage Software Components</a> app
9	DEV	Developer, Business Configuration Expert	Perform the same changes as for the correction in the main branch and release them	ABAP Development Tools for Eclipse  <a href="#">Maintain Business Configurations</a> app  <a href="#">Export Customizing Transports</a> app

## 4.2.6.5.5 Use Case 2: One Development and Correction Codeline in a 5-System Landscape

You can apply this setup if you have permanent/infinite development activities for large applications with many developers, where development cannot be paused to implement an urgent correction. Corrections need to run in parallel to development and on a released state. You need to separate testing from development to ensure the solutions also runs in a non-development system before being delivered to production.



General considerations:

- Systems COR and QAS have the same software state as production system PRD, unless a new change is tested and released. This means transport requests are released in the DEV system only if development is completed and if it is planned to import the changes to the production system.
- Releases are planned and communicated to development in advance:
  - Upon cut-off date, development is finished. All development that is released at this time must be tested and be of good quality. From then on, you have to fix defects in the COR system and maintain them in parallel in the DEV system.
  - Upon release date, all defects must be fixed. If you make the decision during testing in the QAS system that a complete functionality is not delivered, developers must delete, revert, or disable the functionality in the COR system and release the corresponding transport requests. You cannot remove objects from the release branch, e.g. by deselecting transport requests. To revert objects to an older transported state, use the compare editor of the Eclipse *History* view. If the withdrawal of the functionality shall be performed in the DEV system as well it is considered as a correction and you have to perform double maintenance of corrections into the DEV system. The released software state from the COR system is imported into the production system(s) PRD.

- System COR is usually locked for development. First, this means developers cannot do changes by default and there are two approaches how to handle this:

	User Locking	Read-Only + Write Developer Role
How-to Details	Unlock user on demand	Assign write role on demand
Pros	No additional role needed	No generic read user needed  No logon with different user for read access needed  User-specific auditing
Cons	Generic read user needed if you want to provide read access	Additional role needed

Second, developers are also not allowed to create transport requests and tasks on their own, but it's the release manager who creates them for all developers. This separation is achieved by giving business catalog SAP\_A4C\_BC\_TRN\_MNG\_PC (Development - Transport Management) to the release manager instead of the developer role in correction system COR.

## For Go Live/Development after Go Live (Including Deferrable Corrections)

### Starting Situation for Go Live:

Recently created development system DEV and other already existing ABAP systems cannot be based on some branch yet, as the software component does not exist yet.

The Go Live process is characterized by creating different systems only when needed for the first time, however, you can provision the systems already beforehand. Additionally, the future solution's software component does not exist from the start. The resulting release branch is YYYY-01. Apart from this, the Go Live process does not differ from the release development processes.

### Starting Situation after Go Live:

- Development system DEV and test system TST are on the main branch
- Correction system COR, quality assurance system QAS, and production system PRD are on release branch YYYY-<nn>

This process can also be used for deferrable corrections, which do not need to reach production before the next development release. These corrections are handled like normal development.

Step	System	Role	Task	Tool
0	DEV	Release Manager	At Go Live only: create a software component of type <i>Development</i> and, if required, a software component of type <i>Business Configuration</i> . Pull the software component(s) into the system.  If required, create a customizing transport request and create tasks for the relevant business configuration expert(s)	<a href="#">Manage Software Components app</a>  <a href="#">Export Customizing Transports app</a>
1a	DEV	Developer	Develop a new functionality or a deferable correction. All changes are collected in Workbench transport requests	ABAP Development Tools for Eclipse
1b	DEV	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests	<a href="#">Maintain Business Configurations app</a>
2 a	DEV	Developer	Once development is finished, release the transport request(s). The changes are now in the main branch	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>
2 b	DEV	Release Manager	The released API has to be part of development and the development has to be finished.  Snapshots are pulled after development after the API objects are released. The snapshot has to be set to check-relevant. Then, the ATC compatibility check should run.	<a href="#">Manage API Snapshots app</a>
3	TST	Release Manager	Pull the software component(s) into system TST	<a href="#">Manage Software Components app</a>

Step	System	Role	Task	Tool
4	TST	Tester	Test the change and report the test result  If changes are required, repeat steps 1-4	ABAP Development Tools for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
5	QAS or any other	Release Manager	Cutoff: at cutoff date, create a release branch YYYY-<nn+1> for each software component (at Go Live: YYYY-01) (release candidate)	<a href="#">Manage Software Components app</a>
6	QAS	Release Manager	Check out the release branch YYYY-<nn+1> of each software component (at Go Live: YYYY-01) into system QAS	<a href="#">Manage Software Components app</a>
7	QAS	Tester	Test the release candidate and report the test result	ABAP Development Tools for Eclipse with ADT and custom SAP Fiori apps as well as external test tools  External documentation tool
8	COR	Release Manager	Check out the branch YYYY-<nn+1> of each software component (at Go Live: YYYY-01) into system COR	<a href="#">Manage Software Components app</a>
9	COR	Release Manager	Enable the respective development users for development in system COR, depending on the process you decided for, either by unlocking or assigning a different role	<a href="#">Maintain Business Users app</a>

Step	System	Role	Task	Tool
			If required, create a customizing transport request and create tasks for the relevant business configuration expert(s)	<a href="#">Export Customizing Transports app</a>
10a	COR	Developer	Implement development-related corrections	ABAP Development Tools for Eclipse
10b	COR	Business Configuration Expert	Implement corrections to business configuration content	<a href="#">Maintain Business Configurations app</a>
11	COR	Developer	Release the development transport request(s). The changes are now in the release candidate.	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>
12	QAS	Release Manager	Pull the software component(s) to get the correction into the already checked out release branch YYYY-<nn+1> (@Go Live: YYYY-01)	<a href="#">Manage Software Components app</a>
13	QAS	Tester	Test the change and report the test result	ABAP Development Tools for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
			If changes are required, repeat steps 11-13	
14	QAS	Release Manager	Release decision: the changes are successfully tested and approved	External documentation tool
15	PRD	Release Manager	Check out the release branch YYYY-<nn+1> of each software component (at GoLive: YYYY-01) into system PRD	<a href="#">Manage Software Components app</a>

Step	System	Role	Task	Tool
16	COR	Release Manager	Disable the respective development users for development in system COR	<a href="#">Maintain Business Users app</a>
	DEV		If required, create a customizing transport request for the double maintenance of the correction, and create tasks for the relevant business configuration expert(s)	<a href="#">Export Customizing Transports app</a>
17a	DEV	Developer	Perform double maintenance of your development-related corrections in the main branch	ABAP Development Tools for Eclipse
17b	DEV	Business Configuration Expert	Perform double maintenance of your business configuration corrections in the main branch	<a href="#">Maintain Business Configurations app</a>
17c	DEV	Release Manager	Release the transport request(s). The corrections are now implemented both in the main branch and in the current release branch.	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>
18	TST	Release Manager	Pull the software component(s) to get the correction into the already checked out main branch	<a href="#">Manage Software Components app</a>

## Skipping a Release

If issues during the test phase of YYYY-<nn+1> cannot be fixed in a reasonable time frame until the next release date, you can skip that release, especially if you have a tight release schedule ("continuous delivery" model). In that case, you have to perform double maintenance for the unfinished corrections from YYYY-<nn+1> in the main branch of the development ABAP system, release them, and create the new release branch YYYY-<nn+2> derived from that main branch. That way, branch YYYY-<nn+2> contains finished new development as well as the unfinished corrections from branch YYYY-<nn+1>. Afterwards, you can bring system COR and QAS to branch YYYY-<nn+2> and continue with that.

### ⓘ Note

Branches cannot be deleted or marked as obsolete. Therefore, it's important to use other tools to inform consumers about not using branch YYYY-<nn+1>.

## Urgent Corrections

### The starting situation:

- Development system DEV and test system TST are on the main branch
- Correction system COR, quality assurance system QAS, and production system PRD are on release branch YYYY-<nn>

This process is a subset of the previous development process and can be applied to corrections that are too urgent to release with the next development release.

Step	System	Role	Task	Tool
1	COR	Release Manager	Enable the respective development users for development in system COR, depending on the process you decided for, either by unlocking or assigning a different role.	<a href="#">Maintain Business Users app</a>
			If required, create a customizing transport request, and create tasks for the relevant Business Configuration Expert(s)	<a href="#">Export Customizing Transports app</a>
2a	COR	Developer	Create a Workbench transport request and implement the correction	ABAP Development Tools for Eclipse
2b	COR	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests	<a href="#">Maintain Business Configurations app</a>
3	COR	Release Manager	Release the transport request(s). The changes are now in the release candidate	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>
4	QAS	Release Manager	Pull the already checked out branch YYYY-<nn+1> of each software component into system QAS	<a href="#">Manage Software Components app</a> or external tool calling the pull service of communication scenario Test Integration

Step	System	Role	Task	Tool
5	QAS	Tester	Test the change and report the test result  If changes are required, repeat steps 2-5	ABAP Development Tools for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
6	QAS	Release Manager	As the correction was successfully tested before it gets approved now.	External documentation tool
7	PRD	Release Manager	Pull the already checked out release branch YYYY-<nn+1> of each software component into system PRD	<a href="#">Manage Software Components app</a>
8	COR	Release Manager	Disable the respective development users for development in system COR	<a href="#">Maintain Business Users app</a>
	DEV		If required, create a customizing transport request for the double maintenance of the corrections, and create tasks for the relevant business configuration expert(s)	<a href="#">Export Customizing Transports app</a>
9a	DEV	Developer	Perform double maintenance of your development-related corrections in the main branch	ABAP Development Tools for Eclipse
9b	DEV	Business Configuration Expert	Perform double maintenance of your business configuration corrections in the main branch	<a href="#">Maintain Business Configurations app</a>
9c	DEV	Release Manager	Release the transport request(s). The corrections are now implemented both in the main branch and in the current release branch.	ABAP Development Tools for Eclipse: Transport Organizer or <a href="#">Export Customizing Transports app</a>

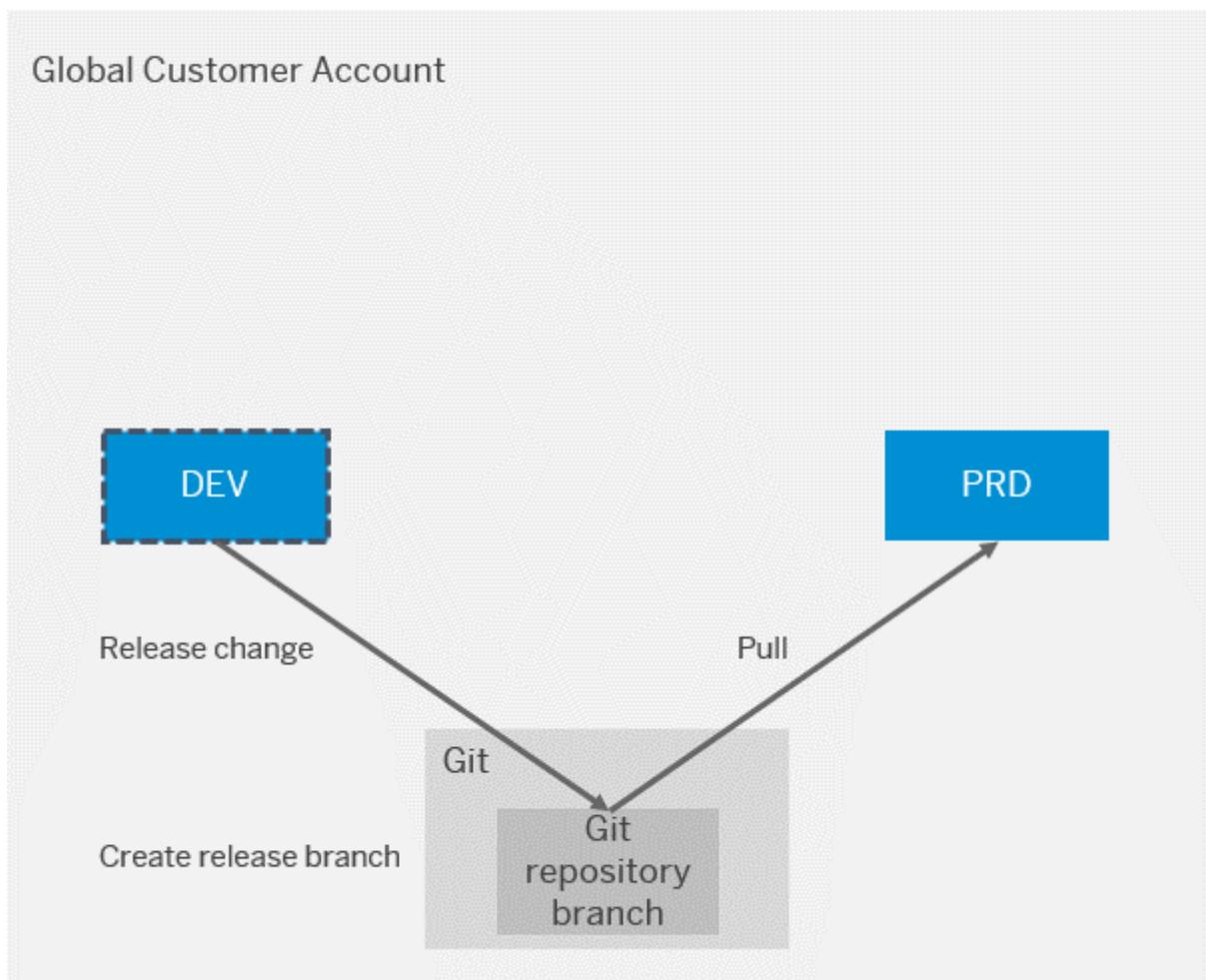
Step	System	Role	Task	Tool
10	TST	Release Manager	Pull the software component to get correction into already checked out main branch	<a href="#">Manage Software Components app</a>

#### 4.2.6.5.6 Use Case 3: One Codeline with On-Demand Development Systems

You can apply this setup if you have:

- Small development projects (for example one SAP Fiori application, one developer)
- Occasional development activities (for example after an initial development phase, it is expected to only implement new features on a yearly basis)

The ABAP system landscape consists of a permanent production system and an on-demand development system.



The advantage of this setup is that you have to administer few systems and only pay for the development system during the development periods. The payment model according to the lifetime of a system requires an SAP BTP enterprise agreement contract.

The disadvantages of this setup are that de-commissioning the development system means losing the change history as well as the configuration and application data.

The user base and authorizations have to be set up each time the development system is provisioned.

Not testing with a transported version of the solution always comes with the risk of forgetting to transport an object. This can be avoided by testing in the development system on the release branch.

A much more production-like test can only be ensured in a non-development system, where the objects of the solution are initially created and changed by pulls, and where there is no authorization automatism for testing, see tester role in [Required Business Roles \[page 739\]](#).

We thus strongly recommend developing with [Use Case 1: One Codeline in a 3-System Landscape \[page 742\]](#) instead of use case 3. The quality assurance system might be de-/commissioned on demand then.

## For the Go Live

### Starting situation:

- A (temporary) development system is set up
- The production system is set up
- Both systems are based on the main branch

Step	System	Role	Task	Tool
1	DEV, PRD	System Administrator	Provision a development and a production system	SAP BTP cockpit
2	DEV, PRD	User Administrator	Create users and maintain roles	<a href="#">Maintain Business Users</a> and <a href="#">Maintain Business Roles</a> app
3	DEV	Release Manager	Create a software component of type <i>Development</i> and, if required, a software component of type <i>Business Configuration</i> . Pull the software component(s) into the development system.	<a href="#">Manage Software Components</a> app
4	PRD	Release Manager	Pull the (empty) software component(s) into the production system	<a href="#">Manage Software Components</a> app

Step	System	Role	Task	Tool
5a	DEV	Developer	Develop new functional requirements or correct existing functionalities. All required changes are collected in Workbench transport requests	ABAP Development Tools for Eclipse
5b	DEV	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests.	<i>Maintain Business Configurations</i> app
6	DEV	Tester	Test the change and report the test results	ABAP Development Tools for Eclipse with and custom SAP Fiori apps as well as external test tools  External documentation tool
		Release Manager	If changes are required, repeat steps 5-6	
7	DEV	Release Manager	Release decision: the changes are successfully tested and approved.	External documentation tool
8	DEV	Release Manager	Release the transport request(s)	ABAP Development Tools for Eclipse: Transport Organizer or <i>Export Customizing Transports</i> app
9	DEV	Release Manager	Create a release branch YYYY-<nn> for each software component	<i>Manage Software Components</i> app
10	PRD	Release Manager	Check out the release branch YYYY-<nn> of each software component	<i>Manage Software Components</i> app
11	DEV	System Administrator	De-provision system DEV	SAP BTP cockpit

## Productive Usage

- The production ABAP system is based on the main branch

- Since no development can be performed in the production ABAP system, a temporary development ABAP system must be set up if occasional development is required

## Occasional Development

- The production ABAP system is based on the main branch
- A temporary development ABAP system is set up
  - All software components are imported

Step	System	Role	Task	Tool
1	DEV	Release Manager	Provision a new development system	SAP BTP cockpit
2	DEV	Release Manager	Create users, maintain and assign roles	<i>Manage Business Users</i> and <i>Manage Business Roles</i> app
3	DEV	Release Manager	Import main branch of each software component into the newly provisioned development system	<i>Manage Software Components</i> app
4a	DEV	Developer	Develop features. All required changes are collected in Workbench transport requests.	ADT for Eclipse
4b	DEV	Business Configuration Expert	Maintain business configuration. All changes are collected in customizing transport requests.	<i>Maintain Business Configurations</i> app
5	DEV	Developer	Release all transport requests	ADT for Eclipse: Transport Organizer or <i>Export Customizing Transports</i> app
6	DEV	Tester	Test new developments/fixes	ADT for Eclipse and custom SAP Fiori apps as well as external test tools
			If changes are required, repeat steps 3-5	
7	PRD	Release Manager	Pull the main branch of each software component into system PRD	<i>Manage Software Components</i> app
8	DEV	Release Manager	Delete the development system	SAP BTP cockpit

## 4.2.6.5.7 Double Maintenance of Corrections into Development

For corrections, double maintenance is necessary so that everything that had to be fixed during release testing or usage can be retrofitted into the development system.

### ⚠ Caution

Missing double maintenance can lead to data loss in production systems.

As there is no selective picking of transport requests to merge corrections back into the main branch, this is manual effort.

You can use the *Compare Editor* in ABAP Development Tools for Eclipse to merge corrections back to the main branch.

The process would be as follows:

You can compare the latest version of an object in your ABAP Development Tools project in a hotfix system with the latest version of the same object in the development system. If there are newly created objects, you have to create an object manually in the development ABAP system.

To make a comparison, from the context menu of the object in the *Project Explorer* or in the source code editor, choose  *Compare With*  *ABAP project*, where ABAP project is one of the ABAP projects defined in ABAP Development Tools.

Step	System	Role	Task	Tool
1	COR	Developer	Select the corrected objects in the transport request and open them.  This can be done selectively object by object.	ABAP Development Tools for Eclipse: Transport Organizer View
2	COR, DEV	Developer	1. Select the resources in the Project Explorer in correction system COR 2. From the resource's pop-up menu, select Compare With 3. Select the ABAP project that points to your development system DEV	ABAP Development Tools for Eclipse: Project Explorer

Step	System	Role	Task	Tool
3	DEV	Developer	Merge the changes into the DEV system	ABAP Development Tools for Eclipse

#### 4.2.6.5.8 Transport of Business Configuration

Business configuration is client-dependent, just like master or transactional data, and after being imported, it is only available in the client where you have executed the import.

You usually maintain business configuration content in a non-productive environment and import it into test and productive environments. To do so, you record business configuration changes on transport requests of type *Customizing*. Once you release the request, all the changes are pushed into the remote repository associated with the used software component, just as for Workbench transport requests. Therefore, you can import business configuration content by pulling that same software component in the target system/client 100.

Depending on your scenario, you can use one of the following transport patterns:

- Isolated tenants – no transport

If you use an isolated tenant, you don't have to clone and transport a software component of type *Business Configuration*. You can still create content but it is not imported or exported. Since a customizing transport request is a technical prerequisite for business configuration changes, the changes are written to a local customizing transport request. That means, you just have to create a transport request. The system then generates a local request and sets its category to *Default*. See [Working in the Export Customizing Transports App \[page 2584\]](#).

##### ⓘ Note

All consumer tenants ( $\geq 200$ ) in multitenant systems are isolated tenants because the administration of software components is not supported in these tenants.

- Simple – transports via software component of type *Business Configuration*. See [Transport Business Configuration via Software Component of Type Business Configuration \[page 761\]](#).  
You can transport business configuration for simple solutions via a software component of type *Business Configuration*. Since you can only clone one software component of this type in each system, all business configuration content is pushed to the same remote repository.  
In this scenario, you have to create and manage customizing transport requests with the *Export Customizing Transports* SAP Fiori app. See [Export Customizing Transports \[page 2582\]](#).
- Multiple software components/Git repositories – transports via software components/Git repositories of type *Development*. See [Transport Business Configuration via Software Components of Type Development \[page 762\]](#).  
If you want to work on multiple separated development projects in parallel in one development system with decoupled release schedules and in multiple software components, it is not advisable to use a software component of type *Business Configuration* because you may need to transport different sets of business configuration content independently from one another. Instead, transport the configuration together with the development via the corresponding software components of type *Development*.

In all of these scenarios you may create and manage customizing transport requests with the *Export Customizing Transports* SAP Fiori app. See [Export Customizing Transports](#).

To use the [Manage Software Components](#) app, business role SAP\_BR\_ADMINISTRATOR has to be assigned.

To use the [Export Customizing Transports](#) app, business role SAP\_BR\_BPC\_EXPERT or SAP\_BR\_ADMINISTRATOR has to be assigned.

1. In the [Manage Software Components](#) app, create a software component of type [Business Configuration or Development](#) according to your scenario if transport of business configuration is required. And clone it to the tenant where you want the business configuration to be created. See [How to Create Software Components](#) and [How to Clone Software Components](#).

#### Note

If you have already started development, such components may already be available.

2. In the [Export Customizing Transports](#) app, create a transport request. See [Working in the Export Customizing Transports app](#). The system automatically sets its category to [Default](#). Any user that is assigned to one of the transport request's tasks can use the request for change recording. In case transport is required, assign the [transport target](#) according to the software component that has been created. An empty transport target results in a local transport and should be used in case of isolated tenants.

#### Restriction

There can only be one open transport request of type [Default](#) at a time.

3. Release the transport request. See [Working in the Export Customizing Transports app](#).

#### Note

In exceptional cases, for example for an urgent preimport that must be performed in parallel to the default request, you can create additional customizing transport requests. To create an additional customizing transport request in the [Export Customizing Transports](#) app, change the transport category from [Default](#) (`SAP_CUS_TRANSPORT_CATEGORY = DEFAULT_CUST`) to [Manual](#) (`SAP_CUS_TRANSPORT_CATEGORY = MANUAL_CUST`).

## 4.2.6.5.8.1 Transport Business Configuration via Software Component of Type Business Configuration

Learn how to transport business configuration via a software component of type [Business Configuration](#).

### Prerequisites

To use the [Manage Software Components](#) app, business role SAP\_BR\_ADMINISTRATOR has to be assigned.

To use the [Export Customizing Transports](#) app, business role SAP\_BR\_BPC\_EXPERT has to be assigned.

## Procedure

1. In the [Manage Software Components](#) app, create a software component of type *Business Configuration* and clone it to the tenant where you want the business configuration to be created. See [How to Create Software Components \[page 2807\]](#) and [How to Clone Software Components \[page 2808\]](#).
2. In the [Export Customizing Transports](#) app, create a transport request. See [Working in the Export Customizing Transports App \[page 2584\]](#). The system automatically associates the request to the previously cloned software component and sets its category to *Default*. Any user that is assigned to one of the transport request's tasks can use the request for change recording.

### **Restriction**

There can only be one open transport request of type *Default* at a time.

3. Release the transport request. See [Working in the Export Customizing Transports App \[page 2584\]](#).

### **Note**

In exceptional cases, for example for an urgent preimport that must be performed in parallel to the default request, you can create additional customizing transport requests. To create an additional customizing transport request in the [Export Customizing Transports](#) app, change the transport category from *Default* to *Manual*. Alternatively, you can use ABAP Development Tools for Eclipse. See [Transport Business Configuration via Software Components of Type Development \[page 762\]](#).

## 4.2.6.5.8.2 Transport Business Configuration via Software Components of Type Development

Learn how to transport business configuration via a software component of type *Development*.

### Prerequisites

To use the [Manage Software Components](#) app, business role `SAP_BR_ADMINISTRATOR` has to be assigned.

To use the [Export Customizing Transports](#) app or to create a customizing request in ABAP Development Tools for Eclipse, business role `SAP_BR_BPC_EXPERT` or administrator role `SAP_BR_ADMINISTRATOR` need to be assigned.

To use the Transport Organizer view in ABAP Development for Eclipse, business role `SAP_BR_DEVELOPER` has to be assigned. See [Transport Organizer](#).

## Procedure

1. In the [Manage Software Components](#) app, create software components of type *Development* and clone them to the tenant where you want the business configuration to be created. See [How to Create Software Components](#) and [How to Clone Software Components](#).

 Note

If you have already started development, such components may already be available.

2. In ABAP Development Tools for Eclipse, use the [Transport Organizer](#) view to create customizing transport requests for each software component, making sure to assign the correct target layer for each of the components used.
3. In the Transport Editor, assign the following attributes to all the created customizing transport requests so that they are displayed in the [Export Customizing Transports](#) app:
  - SAP\_ATO\_TRANSPORT\_TYPE = BC
  - SAP\_CUS\_TRANSPORT\_CATEGORY = MANUAL\_CUST
4. In the [Export Customizing Transports](#) app, release the transport request. See [Working in the Export Customizing Transports App](#).

 Note

When developing custom business configuration objects, you can decide if you want to provide a transport selection screen for change recording. If you are using this transport pattern, we recommend autofilling the transport request based on the software component of the business configuration object.

### 4.2.6.6 Information for Audit

The following information is relevant for audit:

- Git repository: The creation of a Git repository branch creates a clone of the parent branch (including all development objects in the branch)
  - During a pull or checkout, all objects that differ from the previously checked out commit and the pulled commit are imported into the ABAP system. Transport requests cannot be imported selectively into subsequent ABAP systems. Together with the information which branch and commit ID was pulled into an ABAP system, you can reproduce which version of an object was in an ABAP system at which point in time.
  - The content of the Git repository with all its commits and objects is not directly accessible to users of an ABAP system. Only SAP employees have access to it for support purposes because the information is typically not required for an audit. In exceptional cases, the content of the Git repository can be requested via a service request.
- In the Manage Software Components SAP Fiori app and in ABAP Development Tools (ADT), the following information is available for a regular audit.
  - Manage Software Components app:

- Action history of imports (pulls and branch checkouts), which also contains an execution log and a transport log per import. As these two logs are only saved temporarily in the ABAP system, we recommend to save them to an excel if they are needed for error analysis or audit.
- Information on the last commit (commit ID, commit message consisting of the transport request number and description, commit author and last commit timestamp) for each branch
- ADT
  - In the *Transport Organizer*, you can find the transport requests together with the piece list
  - In the *Revision History*, you can find the history of an object

## 4.2.6.7 Automate the Software Lifecycle Management Process

To facilitate your software lifecycle management process, you can automate it by using communication scenario SAP\_COM\_0510 and SAP\_COM\_0735. See [Pulling Git Repositories to an ABAP Environment System \[page 788\]](#) and [Running ABAP Unit Test Runs \[page 765\]](#).

These communication scenarios contain several tools that can be used to craft a continuous integration process.

To make the setup of continuous integration pipelines as easy as possible, the open source project "Piper" was created. In general, there are various functions (Library steps) as well as whole pipelines available for reuse. You can find various steps that implement the functionality of the APIs of SAP\_COM\_0510 and SAP\_COM\_0735.

The reusable continuous integration pipeline contains the following steps:

- Creating an SAP BTP, ABAP environment system
- Setting up a communication arrangement for the APIs
- Cloning a software component / Git repository
- Running ABAP Test Cockpit checks
- Running ABAP unit tests
- Deprovisioning the SAP BTP, ABAP environment system

### Note

You can only dynamically create SAP BTP, ABAP environment systems during the pipeline execution if you are using a consumption-based model. See [What Is the Consumption-Based Commercial Model?](#)

## Related Information

[Project "Piper"](#) 

#### **4.2.6.7.1 Running ABAP Unit Test Runs**

## Prerequisites

- You have an SAP BTP ABAP environment system.
  - You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
  - You've created a *Communication System* as described in [How to Create Communication Systems](#).
  - You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
  - You've selected the communication scenario **SAP\_COM\_0735** for your communication arrangement and have mapped it to your communication system.

## Context

ABAP Unit is the standard tool for running unit tests. In this help topic, you'll learn how to start an ABAP Unit Test Run via REST Service.

For more information about ABAP Unit, see [Ensuring Quality of ABAP Code](#).

## Procedure

1. **Get CSRF token:** The first step serves for the authentication on the server. The response header contains a CSRF token, which is used as authentication for the POST request following in step 2.

## Request

**Authentication Type:** Basic Authentication

**GET** <https://<host.com>:<port>/sap/bc/adt/api/abapunit/runs/00000000000000000000000000000000>

## Headers

KEY	VALUE
accept	application/vnd.sap.adt.api.abapunit.run-status.v1+xml
x-csrf-token	fetch

## Response

## Headers

2. **Start an ABAP Unit Run:** To start an ABAP Unit run, insert the **CSRF token** that was retrieved in the first request in the header parameters.

## Request

**POST** https://<host.com>:<port>/sap/bc/adt/api/abapunit/runs

## Headers

KEY	VALUE
x-csrf-token	<token>
content type	application/vnd.sap.adt.api.abapunit.run.v1+xml

## Body

You can specify arbitrary object sets. See also [Object Sets](#) and [XML Representations of Object Sets](#) for more details.

You can provide additional parameters to control the selection of tests. You can restrict the amount of tests to certain risk level and durations. Furthermore, you can decide to run foreign tests, which are connected via test relations.

```
<?xml version="1.0" encoding="UTF-8"?>
<aunit:run title="My Run" context="AIE Integration Test" xmlns:aunit="http://www.sap.com/adt/api/aunit">
    <aunit:options>
        <aunit:measurements type="none"/>
        <aunit:scope ownTests="true" foreignTests="true"/>
        <aunit:riskLevel harmless="true" dangerous="true" critical="true"/>
        <aunit:duration short="true" medium="true" long="true"/>
    </aunit:options>
    <osl:objectSet xsi:type="unionSet" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:osl="http://www.sap.com/api/osl">
        <osl:set xsi:type="osl:packageSet">
            <osl:package includeSubpackages="true" name="PACKAGE_ONE"/>
        </osl:set>
        <osl:set xsi:type="osl:flatObjectSet">
            <osl:object name="ZCL_TEST_ONE" type="CLAS"/>
            <osl:object name="ZIF_TEST_TWO" type="INTF"/>
        </osl:set>
    </osl:objectSet>
</aunit:run>
```

## Response

## Headers

KEY	VALUE
location	/sap/bc/adt/api/abapunit/runs/{runId}

3. **Tracking the Status of the Test Run:** To track the status of the test run, you can make a GET request using the URI contained in the location header.

#### Request

**GET** `https://<host.com>:<port>/sap/bc/adt/api/abapunit/runs/{runId}`

#### Headers

KEY	VALUE
accept	application/vnd.sap.adt.api.abapunit.run-status.v1+xml

#### Response

While the tests are still running, the returned status will be "In Process".

If all tests have been run, the status "Completed" is returned. You'll find the link `/sap/bc/adt/api/atk/results/<UUID>` at the bottom of the body. Use this link to retrieve the ABAP Unit test results in the next step.

#### Body

```
<?xml version="1.0" encoding="UTF-8"?>
<aunit:run xmlns:aunit="http://www.sap.com/adt/api/aunit" title="MyTitle"
context="MyContext">
  <aunit:progress status="FINISHED" percentage="100" />
  <aunit:executedBy user="JOHNDOE" />
  <aunit:time started="2021-02-12T17:09:19Z" ended="2021-02-12T17:09:22Z" />
  <atom:link xmlns:atom="http://www.w3.org/2005/Atom"
href="/sap/bc/adt/api/abapunit/results/0123821392189313" rel="http://
www.sap.com/adt/relations/api/abapunit/run-result" type="application/
vnd.sap.adt.api.junit.run-result.v1+xml"/>
</aunit:run>
```

4. **Retrieve Results:** To get the ABAP Unit results, use the following request.

#### Request

**GET** `https://<host.com>:<port>/sap/bc/adt/api/abapunit/results/{resultId}`

#### Headers

KEY	VALUE
Accept	application/vnd.sap.adt.api.junit.run-result.v1+xml

#### Response

The results are submitted in the JUnit format.

#### Body

As an example, the result may look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<testsuites tests="3" asserts="2" skipped="1" errors="0" failures="1"
timestamp="2021-01-22T19:17:30Z" time="0.36" executedBy="JOHNDOE"
client="000" system="UIA">
```

```

<testsuite tests="2" asserts="2" skipped="1" errors="0" failures="1"
timestamp="2021-01-22T19:17:30Z" time="0.28 " hostname="ldai3uia"
package="test_cars" name="">
    < testcase asserts="1" time="0.01 " name="order"
classname="fugr:zmg_cars.ltc_finder">
        <failure type="Assert Failure" message="Critical Assertion Error:
'Cds_View: ASSERT_EQUALS'">
            Character string different as of position 4Expected
[CL_AUNIT_TESTABLE_OBJECT=====CP]Actual [CL_CAR=====
Test 'LTC_FINDER->ORDER' in Main Program 'SAPLZMG_CARS'
            Stack: Include: <LZMG_CARST99> Line: <38> (ORDER)
        </failure>
    </ testcase>
    < testcase asserts="1" time="0.01 " name="preview"
classname="fugr:zmg_cars.ltc_finder">
        <skipped message="Missing Prerequisites - Dummy: ABORT">
            Test execution skipped due to missing prerequisites Test
'LTC_FINDER->PREVIEW' in Main Program 'SAPLZMG_CARS'
            Stack: Include: <LZMG_CARST99> Line: <44> (PREVIEW)
        </skipped>
    </ testcase>
    < testcase asserts="0" time="0.01 " name="drive"
classname="fugr:zmg_cars.ltc_finder"/>
</ testsuite>
</ testsuites>

```

## 4.2.6.7.2 Reading and Deploying ATC Configurations

### Prerequisites

- You have an SAP BTP ABAP environment system.
- You've created a *Communication User* and generated a **password** as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).

**ⓘ Note**

The URL displayed for the inbound service of the communication arrangement will be needed for the GET requests described below.

- You have selected the communication scenario **SAP\_COM\_0763** for your communication arrangement and have mapped it to your communication system.

### Context

The communication scenario **SAP\_COM\_0763** offers an OData V4 API for maintenance of ABAP Test Cockpit (ATC) configurations on the business objects SATC\_CI\_CF\_RT\_E, SATC\_CI\_CF\_E, and SATC\_CI\_CF\_P\_V.

Therefore, you have to create a valid communication arrangement as described in the prerequisites before performing the actual test steps.

## Procedure

1. **Get CSRF token:** The first step serves for the authentication on the server. Use the URL you've obtained from the *Communication Arrangement*'s inbound service for the **GET** request. The response header contains an CSRF token, which is used as authentication for the GET request following in step 2.

### Request

**Authentication Type:** Basic Authentication. Enter the name of your *Communication User* (not the technical user ID) and the generated password.

**GET** <URL\_from\_Communication\_Arrangement>

### Headers

KEY	VALUE
x-csrf-token	fetch

### Response

### Headers

KEY	VALUE
x-csrf-token	<token>

2. **Read ATC configurations:** To read ATC configurations, prepare a GET request using the following parameters:

### Request

### Header

KEY	VALUE
x-csrf-token	<token>

- All configurations:  
**GET** <URL\_from\_Communication\_Arrangement>/configuration
- By name:  
**GET** <URL\_from\_Communication\_Arrangement>/configuration?\$filter=conf\_name eq '<CONFNAME>'
- Default configuration:  
**GET** <URL\_from\_Communication\_Arrangement>/configuration?\$filter=is\_default eq true

If you want to look deeper into configuration data (e.g. the priorities of the configuration), you can use expands by adding:

- to all priorities: &\$expand=\_priorities

- filtered only on priorities changed from default\_priority: &\$expand=\_priorities(\$filter=priority ne default\_priority)
3. **Deploy ATC configurations:** To deploy an ATC configuration, prepare a POST request and use the body to provide a json (or XML) representing the configuration you want to deploy.

#### Request

##### Header

KEY	VALUE
content-type	application/json
x-csrf-token	<token>

**POST** <URL\_from\_Communication\_Arrangement>/configuration

- Flat Insert of changed priorities:  
When you pass the json from the `get_configuration` with no expand by creating a valid configuration, the default priorities will also be created.
- Deep Insert of changed priorities:  
You can pass the json from the `get_configuration` with expand to filtered priorities and potentially change the `confname` if desired.

## 4.2.6.7.3 Running ABAP Test Cockpit (ATC) Check Runs

### Prerequisites

- You have an SAP BTP, ABAP environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- You've selected the communication scenario **SAP\_COM\_0901** for your communication arrangement and have mapped it to your communication system.

#### ⚠ Restriction

The communication scenario **SAP\_COM\_0510** is deprecated and should no longer be used.

- You've defined a business user in the additional properties in the communication arrangement. To do so, choose the *Additional Properties* link for ATC Check Run in the *Inbound Services* section. Specify a business user (CB) or select a user from the value help. ATC runs started via the service will be scheduled and run with the specified business user.

**ⓘ Note**

The assigned business user needs development authorization for ADT, which is contained in the business catalog SAP\_A4C\_BC\_DEV\_ATC\_RUN\_PC. In case the ATC is used to run ABAP Unit Tests, further specific authorizations might be required.

When you assign the catalog to a dedicated role, do not forget to remove the restriction to read-only access before saving the role.

## Context

The ABAP Test Cockpit (ATC) is the standard tool for checking the quality of ABAP development objects using static checks and ABAP unit tests. In this help topic, you'll learn how to start an ATC Check Run via REST Service.

For more information about the ABAP Test Cockpit, see [Checking Quality of ABAP Code with ATC](#).

## Procedure

1. **(Get CSRF token)** The first step serves for the authentication on the server. The response header contains an CSRF token, which is used as authentication for the POST request following in step 2.

## Request

**Authentication Type:** Basic Authentication

**GET** https://<host.com>:<port>/sap/bc/adt/api/atc/runs/00000000000000000000000000000000

## Headers

KEY	VALUE
accept	application/vnd.sap.atc.run.v1+xml
x-csrf-token	fetch

## Response

Body

```
<?xml version="1.0" encoding="utf-8"?>
<atc:run status="Not Created" xmlns:atc="http://www.sap.com/adt/atc">
    <atc:progress description="No run was created, no objects had to be
checked"/>
    <atc:phases/>
</atc:run>
```

## Headers

2. **(Start an ATC Check Run)** To start an ATC check run, insert the **CSRF token** that was retrieved in the first request in the header parameters.

## Request

**POST** https://<host.com>:<port>/sap/bc/adt/api/atc/runs?clientWait=false

## Headers

KEY	VALUE
x-csrf-token	<token>
content type	application/vnd.sap.atc.run.parameters.v1+xml

## Body

You can specify components, packages or both. If you specify both components and packages, only those objects will be checked that are assigned to packages that belong to the specified software components.

For the package value, enter the name of the package you want to have checked.

You can provide an ATC configuration as additional parameter that is used instead of the default configuration of your system.

You can also provide an ATC check variant as additional parameter. This is used instead of the default check variant of your ATC configuration, regardless of whether an ATC configuration was explicitly provided via parameter or not.

You can specify arbitrary object sets. See also [Object Sets](#) and [XML Representations of Object Sets](#) for more details.

```
<?xml version="1.0" encoding="UTF-8"?>
<atc:runparameters xmlns:atc="http://www.sap.com/adt/atc"
    xmlns:obj="http://www.sap.com/adt/objectset"
    checkVariant="CHECKVARIANT_NAME"
    objectProvider="OBJECTPROVIDER_NAME"
    configuration="CONFIGURATION_NAME">
    <osl:objectSet xsi:type="unionSet" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:osl="http://www.sap.com/api/osl">
        <osl:set xsi:type="osl:multiPropertySet">
            <osl:property key="softwareComponent" value="Z_MY_COMPONENT" />
        </osl:set>
        <osl:set xsi:type="osl:packageSet">
            <osl:package includeSubpackages="false" name="Z_MY_PACKAGE" />
        </osl:set>
    </osl:objectSet>
</atc:runparameters>
```

## Response

## Headers

KEY	VALUE
location	/sap/bc/adt/api/atc/runs/<UUID>

3. **(Tracking the Status of the Check Run)** To track the status of the check run, you can make a GET request using the URI contained in the location header.

#### Request

**GET** `https://<host.com>:<port>/sap/bc/adt/api/atc/runs/<UUID>`

#### Headers

KEY	VALUE
accept	application/vnd.sap.atc.run.v1+xml

#### Response

While the check is still running, the ATC phases will show the status "In Process" or "Undefined" in the body.

The check run is finished when all ATC phases show the status "Completed". You'll find the link `/sap/bc/adt/api/atc/results/<UUID>` at the bottom of the body. Use this link to retrieve the ATC check run results in the next step.

#### Body

```
<?xml version="1.0" encoding="utf-8"?>
<atc:run status="Completed" xmlns:atc="http://www.sap.com/adt/atc">
    <atc:progress description="Run Completed"/>
    <atc:phases>
        <atc:phase title="Determine Object Keys" status="Completed"
number="1"/>
        ...
        </atc:phases>
        <atom:link href="/sap/bc/adt/api/atc/results/<UUID>" rel="http://
www.sap.com/abap/checks/atc/relations/results/displayid" type="application/
xml" title="Result" xmlns:atom="http://www.w3.org/2005/Atom"/>
    </atc:run>
```

4. **(Retrieve Results)** To get the ATC results, use the following request.

#### Request

**GET** `/sap/bc/adt/api/atc/results/<UUID>`

#### Headers

KEY	VALUE
Accept	application/vnd.sap.atc.checkstyle.v1+xml

#### Response

The results are submitted in the checkstyle format. You can find the checkstyle XSD here: <https://github.com/linkedin/pygradle/blob/master/pygradle-plugin/src/test/resources/checkstyle/checkstyle.xsd>.

## Body

As an example, the result may look like this:

```
<?xml version="1.0" encoding="utf-8"?>
<checkstyle version="1.0">
    <file name="PACKAGE/OBJECT_NAME.OBJECT_TYPE">
        <error message="Hard-coded user name (@User's formatted name)" source="CL_CI_TEST_EXTENDED_CHECK_SEC#0821" line="2" severity="error"/>
    </checkstyle>
```

In this example, the error message shows a hard-coded username. Ideally, the formatted user name corresponds to the developer that edited the object for the last time. In case "last changed by" shouldn't be available for the object, the author is used instead. In case the formatted user name can't be determined, the business user id is displayed as fallback.

### ⓘ Note

In the output, the namespace slashes will be replaced by dashes. That means /NAMESPACE / OBJECT\_NAME will be output as -NAMESPACE-OBJECT\_NAME.

## 4.2.6.7.4 Configuring Remote ATC Using a Central Check System

You can perform remote code analysis in ATC, allowing you to analyze custom code remotely with the latest checks.

When using SAP BTP ABAP environment as the central check system, you will use the `Custom Code Migration` app to scope and analyze custom code in an on-premise SAP system for SAP S/4HANA Cloud Public Edition.

To use this SAP Fiori app, your SAP administrator must enable the app to provide access for the relevant users. In addition, you also need to establish an RFC connection for the Cloud Connector. This enables you to access the on-premise system as the checked system from the ABAP environment as central check system remotely.

## Related Information

[Configuring the Checked System \[page 775\]](#)

[Configuring and Administrating the Cloud Connector \[page 778\]](#)

[Maintaining Communication Arrangements \[page 782\]](#)

[Configuring the ATC Developer Scenario \[page 783\]](#)

[Custom Code Migration App](#)

## 4.2.6.7.4.1 Configuring the Checked System

You need to configure an RFC destination from your on-premise system (checked system) to the SAP BTP ABAP environment as central check system. This enables you to run ATC checks in the central check system for your custom code in the checked system.

### Related Information

[Configuring the User \[page 775\]](#)

[Installing Remote Stubs \[page 777\]](#)

### 4.2.6.7.4.1.1 Configuring the User

In the **checked system**, the RFC user needs the following authorizations:

Authorization Object		
Value	Parameter	Value
S_RFC	ACTVT	16 (Execute)
	RFC_TYPE	FUGR
	RFC_NAME	SCA_REMOTE_DATA_ACCESS SABP_COMP_PROCS_E SYCMAPS_REMOTE SYST S_CODE_INSPECTOR_TESTS SUSG_API
S_RFC	ACTVT	16 (Execute)
	RFC_TYPE	FUNC

Authorization Object		
Value	Parameter	Value
	RFC_NAME	FUNCTION_EXISTS
		REPOSITORY_ENVIRONMENT_ALL
	RFC_GET_NAMETAB	
		SVRS_GET_VERSION_DIRECTORY_46
	RFCPING	
		SLINRMT_RUN
	TRINT_PROGRESS_INDICATOR	
		TRINT_TP_UPDATE_TPSTAT
	SLDAG_GET_SYSTEM_NAME	
	CVA_BSP_RUN	
		IDOCTYPE_COLLECT_ATC_INFOS
	IQAPI_QUERY_CONTENT	
		IQAPI_INFOSET_CONTENT
	IQAPI_CHECK_INFOSET	
S_DEVELOP	ACTVT	03 (Display)
	DEVCLASS	*
	OBJNAME	*
	OBJTYPE	*
	P_GROUP	*
S_SYS_RWBO	ACTVT	01 (Create or generate) 02 (Change) 03 (Display)
	DOMAIN	*
	DESTSYS	*
	TTYPE	TRAN
S_TRANSPRT	ACTVT	01 (Create or generate) 02 (Change) 03 (Display)
	TTYPE	TRAN
S_DATASET	ACTVT	34 (Write)

Authorization Object		
Value	Parameter	Value
	FILENAME	*
	PROGRAM	SAPLSABC
		SAPLSTRF

In the **Central Check System**, you need the following user to use transaction ATC to perform custom code checks:

User Role	Description
SAP_SATC_ADMIN	Authorization for setting up ABAP Test Cockpit (ATC) for central quality checking

In addition, you need the following authorization object for importing the Simplification Database into the Central Check System:

Name of Authorization Object	Name of the Authorization Field	Value of the Authorization Field	Description
S_YCM	SYCM_AREA	SDB	Authorization for importing the Simplification Database
	ACTVT	UL	

## 4.2.6.7.4.1.2 Installing Remote Stubs

During ATC execution, the central check system accesses the systems in your landscape remotely through so-called remote stubs using RFC connection. Thus, remote stubs serve as an interface between the central check system and on-premise systems; they return a model of custom code that needs to be checked.

### Prerequisites

In the on-premise system, implement SAP Note [2599695](#) (Custom Code Migration Fiori App: Remote Stubs for the Checked System) in advance.

### Context

You want to analyze your custom code in your on-premise system using the Custom Code Migration app.

## Procedure

1. In the on-premise system, perform transaction SNOTE.
  2. Download SAP Note [2599695](#).
- The *Note Assistant: Note Download* dialog is opened.
3. To implement the remote stubs, select and confirm the displayed SAP Note.
  4. It is also recommended to apply SAP Notes in checked systems. Update your check system by following these steps:
    - a. Perform transaction SNOTE.
    - b. In the menu bar, select *Goto > Other tools > Launch Note Analyzer*.
    - c. Upload the file *SAPNote\_2436688\_Checked\_System.xml* available as attachment in SAP Note [2436688](#). You can now process the notes using SAP Note Analyzer.

### ⓘ Note

In case you don't want to use the Note Analyzer, please also apply the following SAP Notes:

- [2485231](#) - Remote ATC Checks of Modifications and Enhancements
- [2270689](#) - RFC Extractor for performing static checks
- [2190065](#) - ATC/CI: Remote Code Analysis - Object Provider Stub

## Results

The Custom Code Migration app now has access to the on-premise system. You can now start the custom code analysis.

### 4.2.6.7.4.2 Configuring and Administrating the Cloud Connector

The *Cloud Connector* enables the connection between the SAP BTP ABAP environment and on-premise systems in a secured network.

## 4.2.6.7.4.2.1 Connecting the Cloud Foundry Environment Subaccount to the Cloud Connector

### Prerequisites

- You have created a subaccount for the Cloud Foundry environment.
- You have installed the Cloud Connector.

### Context

You want to add and connect your subaccount for the Cloud Foundry environment to the Cloud Connector.

For more information, see

- [Managing Subaccounts](#)
- [Cloud Connector](#)

### Procedure

1. Go to your *Cloud Connector* and select the *Connector* navigation pane.
2. Choose the *Add Subaccount* button in the title toolbar.
3. In the *Add Subaccount* wizard, enter the *Region* of your server. Make sure that you choose the exact same region as in your Cloud Foundry subaccount.
4. Enter the ID of the *Subaccount* that you have created for the Cloud Foundry environment in the *SAP BTP Cockpit*. You will find the subaccount ID under *SAP BTP Cockpit* *your Global Account* *your Cloud Foundry Subaccount* *Subaccount Details* .
5. *Cloud Connector*, enter the [Optional:] To label the subaccount in the *Display Name*.
6. Enter the login email address of the *Subaccount User*.
7. Enter the *Password* of the subaccount user.
8. [Optional: To overwrite the default location ID of the data center in the Cloud Foundry environment, enter the relevant *Location ID*.]
9. [Optional:] Provide a *Description* for further details.
10. To add your subaccount to the *Cloud Connector*, choose *Save*.

## Results

A new entry for the Cloud Foundry subaccount is added in the *Subaccount Dashboard* table in the Cloud Connector.

### 4.2.6.7.4.2.2 Configuring the Access Control (RFC) for the Remote Connection

The resource accessible represents the on-premise system that can be accessed through the *Cloud Connector*.

## Context

You want to enable the connection for SAP BTP ABAP environment to an on-premise system.

For more information, see [Configure Access Control \(RFC\)](#)

## Procedure

1. In the *Cloud Connector*, select the ► [Display Name of the Cloud Foundry Subaccount] ► *Cloud to On-Premise* ▶ navigation pane.
2. To add a new system mapping, choose + from the section toolbar of the *Mapping Virtual To Internal System* table.
3. In the *Add System Mapping* wizard, provide the following data:
  - a. Choose *ABAP System* from the *Back-end Type* drop-down list box.
  - b. Choose *Next*.
  - c. Choose *RFC* as *Protocol* from the drop-down listbox.
  - d. Choose *Next*.

4.  Note

To use a connection **with** load balancing, skip this step and proceed with the next step.

To use a connection **without** load balancing, proceed as follows:

- a. Choose the *Without load balancing (application server and instance number)* radio button.
- b. Choose *Next*.
- c. Enter the following connection details of the on-premise system:

 Note

You will find these details in the SAP Logon of your On-Premise system.

- *Application Server*.
  - *Instance Number* of your Application Server ABAP.
  - Name of the *SAProuter*, if applicable.
- d. Enter a virtual name for the *Application Server*.
  - e. Enter a virtual *Instance Number*.
  - f. Choose *Next*.
  - g. Optional: Enter a description.
  - h. Choose *Next*.

## 5. Note

If you use a connection **without** load balancing, skip this step. Follow the instructions in the previous step instead.

To use a connection **with** load balancing, proceed as follows:

- a. Choose the *With load balancing (system ID and message server)* radio button.
- b. Choose *Next*.
- c. Enter the following connection details of the on-premise system:

## Note

You will find these details in the SAP Logon of your On-Premise system.

- *Message Server*.
  - *System ID*.
  - Name of the *SAProuter*, if applicable.
- d. Choose *Next*.
  - e. Define and enter a virtual name for the *Message Server*.
  - f. Define and enter a virtual *System ID*.
  - g. Optional: Enter a description.
  - h. Choose *Next*.
6. Optional: You can mark the checkbox *Check Internal Host* to immediately check the connection during creation.
  7. To trigger creation, choose *Finish*.

You have added the relevant on-premise system as resource accessible.

Now, you want to define the permitted function modules as resources for a specific back-end system.

To do this, proceed as follows:

8. In the *Cloud To On-Premise* navigation pane, select the relevant system mapping in the *Mapping Virtual to Internal System* section.
9. To define which function modules are executable in the on-premise, choose  *Import resources belonging to a scenario* from the section toolbar in the *Resources Of ...* section.

## Note

To get the relevant import file for the Custom Code Migration scenario, see SAP Note [2861842](#) - Custom Code Migration in SAP BTP ABAP environment: Set up SAP Cloud Connector.

10. Browse the *Scenario File* and choose *Select*.

## Results

You have defined the on-premise system(s) that are accessible from the ABAP environment.

Now, you can call the defined function modules in the on-premise system.

### 4.2.6.7.4.3 Maintaining Communication Arrangements

A communication arrangement describes a communication scenario with a remote system during configuration time. It provides the required metadata for the service configuration.

#### Prerequisites

To create these communication arrangements, you need the `Administrator` role.

You have created a destination to your on-premise system in the *Cloud Connector* that is connected to your BTP subaccount.

#### Context

You want to enable communication from your ABAP environment to your on-premise systems using Remote Function Calls (RFC).

To do this, you need to use the communication arrangement `SAP_COM_0464` (SAP Custom Code Migration Integration).

#### Procedure

1. Log on to the *SAP Fiori launchpad* of your ABAP environment.
2. In the *Communication Management* section, select the *Communication Arrangements* tile.
3. Choose *New*.
4. Use the value help to choose the `SAP_COM_0464` scenario.
5. Define and enter an *Arrangement Name*.

If not yet available or defined, create a new communication system for the communication arrangement to define an endpoint for your checked system.

- a. For the *Communication System*, choose *New*.
  - b. In the *New Communication System* dialog, enter the *System ID* and *System Name* of the checked system.
  - c. Choose *Create*.
  - d. In the *Technical Data* tab under *General*, enter the virtual host as specified in your *Cloud Connector* as *Host Name*. The field *Port* has already been filled in automatically with the default **443**.
  - e. Turn on the slider for *Cloud Connector*.
  - f. Filling in the field *SCC Location ID* is optional.
  - g. Under *RFC Settings*, fill in the fields *Client*, *Instance Number* and *Target Host* as specified as virtual host in your *Cloud Connector*.
  - h. Under *User for Outbound Communication*, choose **+** to assign the RFC user you created in the checked system to the communication system.
  - i. Choose *Create* and then *Save* to save the communication system and to be navigated back to the Communication Arrangement creation page.
6. Under *Additional Properties*, fill out the fields *Object Provider* and *System Group* (both freely selectable). This is the name you then select as your *Object Provider to Remote System* for your custom code migration project in the *Custom Code Migration* app.
  7. Under *Outbound Services* *Retrieve Custom Code* , ensure that the *Service Status* is set to *Active*.
  8. Choose *Save* to save the communication arrangement. A message should now pop up at the bottom of the screen informing you that the activation was successful.

## Results

You can now select the communication arrangement in the *Object Provider to Remote System* field in the *Custom Code Migration* app to establish the connection to your on-premise system.

### 4.2.6.7.4.4 Configuring the ATC Developer Scenario

You can use a system in SAP BTP ABAP environment as a central check system to run ATC checks from an on-premise system against this system (ATC Developer Scenario).

## Prerequisites

- You've defined communication arrangement **SAP\_COM\_0464** (SAP Custom Code Migration Integration) to enable communication from your ABAP environment to your on-premise systems using remote function calls (RFC). For more information, see [Maintaining Communication Arrangements \[page 782\]](#).
- You've implemented [SAP Note 3358660 - Developer Scenario Cloud](#) in your checked system.

## Context

To enable the ATC Developer Scenario, you've to define communication arrangement SAP\_COM\_0936 for your cloud system.

## Procedure

1. Log on to the [SAP Fiori launchpad](#) of your ABAP environment.
2. In the [Communication Management](#) section, select the [Communication Arrangements](#) tile.
3. Choose [New](#).
4. Use the value help to choose the SAP\_COM\_0936 scenario.
5. Define and enter an [Arrangement Name](#).
6. If not yet available or defined, create a new communication system for the communication arrangement to define an endpoint for your checked system.
  - a. For the [Communication System](#), choose [New](#).
  - b. In the [New Communication System](#) dialog, enter the [System ID](#) and [System Name](#) of the checked system.
  - c. Choose [Create](#).
  - d. In the [Technical Data](#) tab under [General](#), select the [Inbound Only](#) check box.  
Be aware that you should only select the [Inbound Only](#) check box if you are using a new communication system. Please do not check this box when you are re-using a communication scenario.
7. In the Communication Arrangement in the [Inbound Communication](#) tab, enter the [User Name](#) you've previously defined as described in step 6.
8. In the [Additional Properties](#) tab, select the [Object Provider](#) you've defined in SAP\_COM\_0464.
9. Choose [Save](#).
10. In the [Code Inspector](#) section, enter the [Reference Check System](#). This is the RFC connection to your cloud system. If not yet available or defined, create a new RFC destination in transaction SM59. To do this, follow these steps (see also [Maintaining Remote Destinations](#)):

### ⓘ Note

In order to setup your RFC connection, you first need to open a **service channel** (on-premise to cloud connection) in your respective Cloud Connector. See [Configure a Service Channel for RFC](#) to learn how to create one.

- a. Choose connection type "3" (ABAP Connection) to connect to your cloud system via [Cloud Connector](#). Connection type "W" (WebSocket RFC) isn't yet supported for the developer scenario
- b. In the [Technical Settings](#) tab, enter the host of your Cloud Connector as [Target Host](#). As instance number, choose the instance number that you have defined for your service channel.

- c. In the *Logon & Security* tab, enter the correct client of your cloud system. The user to be entered here is the inbound communication user that you have assigned to communication scenario SAP\_COM\_0936. Here you need to enter the **alias name** of this user. However, be aware that it cannot exceed 12 characters.
  - d. As an optional step, you can run a connection test, which should be successful.
  - e. Save your destination.
11. In your checked system, run transaction ATC.

The *ABAP Test Cockpit* is opened.
  12. You can now proceed and maintain your destination in *ATC* *Basic Settings* as *Reference Check System*.
  13. Choose *ATC Administration* *Setup* *Basic Settings*.
  14. Go to transaction SCI and create a proxy variant (as described in step 2 in [Configuring the Checked System](#)).
- You can reference one of the SAP delivered ATC variants in your cloud system, for example `LIN_SEC`, or one of your own variants that you've created in the cloud system. Go to [Creating ATC Check Variants](#) for guidance on how to create ATC check variants.
15. Go back to transaction ATC. In the *Basic Settings*, enter the name of this proxy variant as *Global Check Variant*.
  16. Confirm.

## Results

You've enabled a cloud system as the central check system for remote ATC checks from an on-premise system against the cloud system. In SAP GUI or ABAP development tools for Eclipse, you can now run ATC checks in your checked system. In ABAP development tools for Eclipse in the *ATC Problems* view, you can create exemptions that you can then manage in the [Approve ATC Exemptions](#) app.

### 4.2.6.7.5 (Deprecated) Test Integration (SAP\_COM\_0510)

Communication scenario SAP\_COM\_0510 enables you to create continuous integration pipelines for SAP BTP, ABAP environment systems.

You can use this communication scenario to create continuous integration processes. See [Automate the Software Lifecycle Management Process \[page 764\]](#).

#### Note

Please be aware that this communication scenario is deprecated. From now on, please refer to the new communication scenario [API for Managing Git repositories \(0948\)](#).

## 4.2.6.7.5.1 API to Manage Git Repositories

You can use the `MANAGE_GIT_REPOSITORY` API to pull or clone Git repositories to ABAP environment systems and to create and checkout branches. The `MANAGE_GIT_REPOSITORY` API belongs to the Communication Scenario SAP\_COM\_0510.

### ⓘ Note

In ABAP environment, Git repositories are wrapped in software components. These are currently managed in the **Manage Software Components** app. The technical parameter `sc_name` used in this API needs to be the name of the Git repository on the ABAP environment system – the name of the software component.

### ⓘ Note

It is currently not possible to execute/trigger multiple actions (clones/pulls/deletes) in parallel on one system, regardless of the fact that the calls run asynchronously.

## Related Information

[Cloning Git Repositories to an ABAP Environment System \[page 786\]](#)

[Pulling Git Repositories to an ABAP Environment System \[page 788\]](#)

[Working with Branches on a Git Repository \[page 792\]](#)

## 4.2.6.7.5.1.1 Cloning Git Repositories to an ABAP Environment System

### Prerequisites

- You have an ABAP environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- You have selected the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.

## Context

If a software component is not in your system yet, you will first need to clone it once to import it into your system. Please note that you can clone a software component only once into your system.

### ⓘ Note

In ABAP environment, Git repositories are wrapped in software components. These are currently managed in the [Manage Software Components](#) app. The parameter `sc_name` passed to this API needs to be the name of the Git repository on the ABAP environment system – the name of the software component.

## Procedure

1. **(Authentication on the server)** The first step serves the authentication on the server. The response header contains an x-csrf-token, which is used as authentication for the POST request following in [step 2](#).

### Request

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: fetch
```

### Response

```
HTTP/1.1 200 OK
X-csrf-token: xCsrfToken
```

2. **(Cloning a Git repository)** To trigger the clone, you need to insert the x-csrf-token that was retrieved in the first request in the header parameters. The Git repository and the branch you want to clone are passed in the body of the request. If you don't enter a branch name, the main branch is automatically selected. Alternatively you can clone a specific tag by entering the tag name. Make sure that the tag exists on the branch. If that's not the case, no clone is executed.

### Request

```
POST /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json

{
  "sc_name" : "/DMO/GIT_REPOSITORY",
  "branch_name": "main",
  "tag_name": "my-release-tag"
}
```

3. **(View all clones)** To view all clones that have been made, you can make a GET request.

### Request

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones HTTP/1.1
Host: host.com
```

```
Authentication: basicAuthentication
Accept: application/json
```

## Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='02ceab7d-ff04-1eda-blea-eb7fdbf28bc4',sc_name='ZHM_INITIAL_TEST',branch_name='')",
          "uri": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='02ceab7d-ff04-1eda-blea-eb7fdbf28bc4',sc_name='ZHM_INITIAL_TEST',branch_name='')",
          "type": "cds_sd_a4c_a2g_gha_sc_web_api.ClonesType"
        },
        "uuid": "02ceab7d-ff04-1eda-blea-eb7fdbf28bc4",
        "sc_name": "/DMO/GIT_REPOSITORY",
        "branch_name": "",
        "import_type": "Clone",
        "namespace": "",
        "status": "S",
        "status_descr": "Success",
        "criticality": 3,
        "user_name": "administrator@example.com",
        "start_time": "/Date(1594898863000+0000)/",
        "change_time": "/Date(1594898891000+0000)/"
      },
      // all clones for this system instance will be listed here ...
    ]
  }
}
```

## 4.2.6.7.5.1.2 Pulling Git Repositories to an ABAP Environment System

### Prerequisites

- You have an ABAP environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- You have selected the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.
- You have cloned the Git repository to the ABAP Environment system. See [Cloning Git Repositories to an ABAP Environment System \[page 786\]](#).

## Context

You can use the communication scenario [SAP\\_COM\\_0510](#) to pull Git repositories to an ABAP environment system.

### ⓘ Note

In ABAP environment, Git repositories are wrapped in *Software Components*. These are currently managed in the App [Manage Software Components](#). The parameter passed to this API needs to be the name of the Git repository on the ABAP environment system – the name of the Software Component.

## Procedure

1. **(Authentication on the Server)** The first step serves the authentication on the server. The response header contains an x-csrf-token, which is used as authentication for the POST request following in [step 2](#).

### Request

```
GET/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: fetch
```

### Response

```
HTTP/1.1 200 OK
X-csrf-token: xCsrfToken
```

2. **(Pull a Git Repository)** To trigger the pull of a Git repository, you have to insert the *x-csrf-token* that was retrieved in the first request in the header parameters. The Git repository you want to pull is passed in the body of the request.

### Request

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
{
  "sc_name" : "/DMO/GIT_REPOSITORY"
}
```

### Response

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')",
      "uri": "https://host.com/sap/opu/
```

```

odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')", "type":  

"cds_sd_a4c_a2g_gha_sc_web_api.PullType"  

}  

,  

"uuid": "UUID ",  

"sc_name": "/DMO/GIT_REPOSITORY ",  

"namespace": "",  

"status": "R",  

"status_descr": "Running",  

"start_time": "/Date(1571227437000+0000)/",  

"change_time": "/Date(1571227472000+0000)/",  

"criticality": 2,  

"user_name": "CC0000000001",  

"to_Execution_log": {  

    "__deferred": {  

        "uri": "https://host.com/sap/opu/odata/sap/  

MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log"  

    }
}  

,  

"to_Transport_log": {  

    "__deferred": {  

        "uri": "https://host.com/sap/opu/odata/sap/  

MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log"
    }
}
}
}

```

3. **(Tracking the Status of the Pull)** To track the status of the pull, you can make a GET request using the uuid contained in the response. You can also read the URI directly from the “\_\_metadata” of the response.

#### Request

```

GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID') HTTP/1.1  

Host: host.com  

Authentication: basicAuthentication  

Accept: application/json

```

The response contains the same entity as the second request. The Pull is successful, when the “status” in the response body has the value **S**. The status description “status\_descr” will then return **Successful**. In case of an error “status” will have the value **E** and “status\_descr” the value **Error**.

4. **(Retrieving Logs)** To get the Execution Log and the Transport Log after the Pull is finished, you can use the following requests. Alternatively, you can use the URIs from the response of the POST request. You can also check both logs in the *Manage Software Components* app for better readability.

#### For the Execution Log:

#### Request

```

GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/  

to_Execution_log HTTP/1.1  

Host: host.com  

Authentication: basicAuthentication  

Accept: application/json

```

#### Response

```

HTTP/1.1 200 OK  

Content-Type: application/json

```

```
{

```

```

    "d": {
      "results": [ {
        "__metadata": {
          "id": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
ExecutionLogs(index_no=1m,uuid=guid'UUID')",
          "uri": "host.com/sap/opu/
odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')",
          "type": "cds_sd_a4c_a2g_gha_sc_web_api.ExecutionLogsType"
        }
      },
      {
        "index_no": "1",
        "uuid": "UUID",
        "type": "Information",
        "descr": "Step X: MESSAGE",
        "timestamp": "/Date(1571227438000+0000)/",
        "criticality": 0,
      }
    ]
  }
}

```

### For the Transport Log:

#### Request

```

GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/
to_Transport_log HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json

```

#### Response

```

HTTP/1.1 200 OK
Content-Type: application/json

```

```

{
  "d": {
    "results": [ {
      "__metadata": {
        "id": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
ExecutionLogs(index_no=1m,uuid=guid'UUID')",
        "uri": "host.com/sap/opu/
odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')",
        "type": "cds_sd_a4c_a2g_gha_sc_web_api.ExecutionLogsType"
      }
    },
    {
      "index_no": "1",
      "uuid": "UUID",
      "type": "Information",
      "descr": "Step X: Message",
      "timestamp": "/Date(1571227438000+0000)/",
      "criticality": 0,
    }
  ]
}

```

## 4.2.6.7.5.1.3 Working with Branches on a Git Repository

### Prerequisites

- You have an ABAP environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- You have selected the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.
- You have a Git repository/ software component with a main branch. A main branch is created when you push content to the repository for the first time, i.e. you release a transport request for the repository/ software component.

### Context

You can use the communication scenario SAP\_COM\_0510 to work with Git repositories on an SAP BTP, ABAP environment system.

#### ① Note

In ABAP Environment, Git repositories are wrapped in software components. These are currently managed in the [Manage Software Components](#) app. The technical parameter `sc_name` used in this API needs to be the name of the Git repository on the ABAP Environment system – the name of the software component.

### Procedure

1. **(Authentication on the Server)** The first step serves the authentication on the server. The response header contains an x-csrf-token, which is used as authentication for the POST request following in [step 2](#).

#### Request

```
GET/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: fetch
```

#### Response

```
HTTP/1.1 200 OK
X-csrf-token: xCsrfToken
```

2. **(Clone a Git Repository)** If a software component is not yet in your system, you first need to clone it once to import it into your system. For more information on how to do this, see [Cloning Git Repositories to an ABAP Environment System \[page 786\]](#).
3. **(Pull a Git Repository)** In order to work with branches, the Git repository needs to be pulled to the system. For more information on how to do this, see [Pulling Git Repositories to an ABAP Environment System \[page 788\]](#).
4. **(Create a Branch)** To create a branch, you perform a POST request on the `Branches` entity. In the header parameters, you should include the `x-csrf-token` that was retrieved in the first request. Fill in the following data in your request body:
  1. `"sc_name"`: the name of your Git repository
  2. `"branch_name"`: the name of your new branch
  3. `"derived_from"`: the branch that your new branch should derive from

#### Request

```
POST /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
{
  "sc_name" : "/DMO/GIT_REPOSITORY",
  "derived_from" : "main",
  "branch_name" : "newBranch"
}
```

#### Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.BranchesType"
    },
    "branch_name": "newBranch",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "is_active": false,
    "criticality": 0,
    "derived_from": "main",
    "created_by": "CC0000000001",
    "created_on": "/Date(1584967657000+0000)/",
    "last_commit_on": "/Date(1584634658000+0000)/"
  }
}
```

5. **(GET the Branch)** To read the branch entity, you can use the following request:

#### Request

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
Branches(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY' HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.BranchesType"
    },
    "branch_name": "newBranch",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "is_active": false,
    "criticality": 0,
    "derived_from": "main",
    "created_by": "CC0000000001",
    "created_on": "/Date(1584967657000+0000)/",
    "last_commit_on": "/Date(1584634658000+0000)/"
  }
}
```

### ⓘ Note

Please note that the newly created branch is not yet active.

6. **(Checkout a Branch)** Simply creating a branch did not change the state of the system. In order to work with the newly created branch, you need to import the branch to the system. This action is called "checkout branch". You need to pass the Git repository and the name of the branch as query parameters.

## Request

```
POST /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/checkout_branch?
branch_name='newBranch'&sc_name='/DMO/GIT_REPOSITORY' HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
```

## Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')",
      "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.PullType"
    },
    "uuid": "UUID",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "namespace": "",
    "status": "R",
    "status_descr": "Running",
    "start_time": "/Date(1571227437000+0000)/",
    "change_time": "/Date(1571227472000+0000)/",
    "criticality": 2,
    "user_name": "CC0000000001",
```

```

        "to_Execution_log": {
            "__deferred": {
                "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log"
            }
        },
        "to_Transport_log": {
            "__deferred": {
                "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log"
            }
        }
    }
}

```

Behind the scenes, a `Pull` entity is created. It can be used to track the status of the checkout (see steps 3 and 4 in [Pulling Git Repositories to an ABAP Environment System \[page 788\]](#)). This entity is specified in the `__metadata` of the response body.

7. **(Delete a Branch)** You can delete existing branches. Keep in mind, however, that neither the current active branch nor the main branch can be deleted. Both parameters are mandatory.

#### Request

```

DELETE /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches(branch_name='my-to-
be-deleted-branch',sc_name='/DMO/GIT_REPOSITORY') HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json

```

#### Response

Since you're using a DELETE request, an HTTP 204 code is returned as confirmation that your branch has successfully been deleted.

```
HTTP/1.1 204 No Content
```

## 4.2.6.7.6 Software Assembly Integration (SAP\_COM\_0582)

The communication scenario SAP\_COM\_0582 enables partners to assemble a software product via integration in Build pipelines for SAP BTP, ABAP environment systems. With the APIs belonging to this communication scenario, you can assemble a software package, upload the assembly and export logs to the Build pipeline, and upload the software package for registration in the Add-on Assembly Kit as a Service (AAKaaS).

To make the setup of Build pipelines as easy as possible, the open source project "Piper" was created. In general, there are various functions ("Library steps"), as well as whole pipelines, available to reuse. There, you can find the step **abapEnvironmentAssemblePackages** (see [abapEnvironmentAssemblePackages](#) ), that implements the functionality of the APIs belonging to SAP\_COM\_0582. The step **abapEnvironmentAssemblePackages** is embedded in a sequence of steps which are needed to produce and distribute software packages, see the Library steps on [Project "Piper"](#) and [Build and Publish Add-on Products on SAP BTP ABAP Environment](#).

Additionally, there is an example pipeline for a Build process (see [ABAP Environment Pipeline](#) ) that comprises the following:

- performing the Addon.yml and AAKaaS steps

- creating a SAP BTP, ABAP environment system
- setting up Communication Arrangement for the APIs
- importing a software component / Git repository
- building software package
- registering the software package in AAKaaS
- deprovisioning the SAP BTP, ABAP environment system

Please find the documentation regarding project "Piper" here: [Project "Piper"](#)

## 4.2.6.7.6.1 Assembling a Software Package

### Prerequisites

- You have fulfilled all prerequisites mentioned in [Build and Publish Add-on Products on SAP BTP ABAP Environment](#).
- Ideally, you have set up and configured a pipeline with stages and steps as recommended in [ABAP Environment Pipeline](#), and [Configuration](#).

In doing so, the following activities are performed automatically:

- You have an ABAP environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- You have selected the communication scenario [SAP\\_COM\\_0582](#) for your communication arrangement and have mapped it to your communication system.
- You have pulled a repository to the SAP BTP, ABAP environment system using communication scenario [SAP\\_COM\\_0510](#).
- You have provided information on build essentials in an addon.yml file which has already been validated by the Add-On Assembly Kit as a Service (AAKaaS).

### Procedure

1. During the build pipeline run, the step **abapEnvironmentAssemblePackages** is executed (see [abapEnvironmentAssemblePackages](#)). This step runs the assembly of a list of provided installations, support packages or patches in an SAP BTP, ABAP environment system and saves the corresponding SAR archive to the pipeline.

2. An example for the configuration in config.yaml as well as input arguments and values can be found here: [Examples](#).

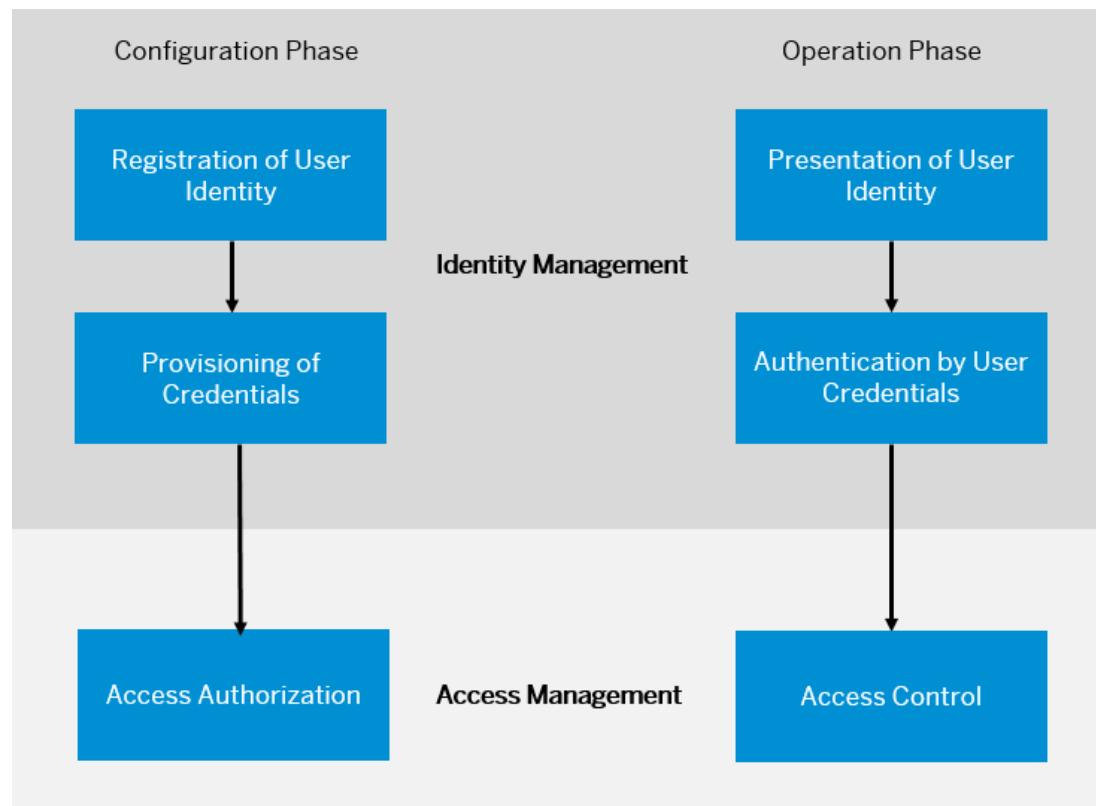
## 4.2.7 Identity and Access Management (IAM)

Learn more about the basic concepts and procedures for managing users and defining authorizations for business services and SAP Fiori applications.

### What Is Identity and Access Management?

The main components of Identity and Access Management are:

- User and Identity Management
- Roles and Authorization
- Authentication



In the configuration phase, an administrator registers and authorizes access rights.

In the operation phase, business users identify and authenticate themselves. The access to applications is based on previously authorized access rights.

In a business scenario, this means that an administrator manages the authentication and authorization of business users, including configuring [trust and identity federation](#) in the subaccount usually by configuring a

custom identity authentication service tenant [page 202]. As a fallback, SAP ID service is configured as the default identity provider. Additionally, an administrator also creates roles using templates, adds roles to role collections, and assigns role collections to business users. The business users can then log on to SAP Fiori applications and get authenticated.

## Who Is This Guide For?

If you're an SAP customer **developer** or partner developer and you create business services in the ABAP environment using ABAP Development Tools for Eclipse, this guide is for you. It guides you through an example of a business service based on a service binding. Note that http services work in a similar way.

In addition, the guide also covers activities that need to be performed by an **administrator**, such as assigning users to business roles.

## What Is This Guide About?

The guide starts with an overview of the basic concepts of identity and access management. It then leads you step by step through the design and implementation of your authorization model using an example.

Why should you care about access control and authorizations? You must know that every new service is automatically available to all developers in a development system, provided the system administrator has assigned them a role that is derived from the developer role SAP\_BR\_DEVELOPER (which is typically the case). However, business users in nondevelopment systems can only access the business service if a developer has defined the appropriate authorizations and an administrator grants the business users access for each relevant system. The same is true for communication users that are needed in scenarios where you work with an API business service (inbound or outbound) from or into the ABAP environment.

The **Identity Management** section gives you an overview about Identity and User Management, which includes identity federation, user types, and user provisioning.

The **Access Management** section includes the following main scenarios about how you can implement and provide authorizations for your services:

- In chapter [Providing \(Unrestricted\) Access for Business Users \[page 815\]](#), you learn how to provide users in a nondevelopment system access to the service.  
Providing unrestricted access is the simplest scenario with only a few steps. The access is unrestricted, though, which means, for example, that you can't differentiate between write and read access.
- In chapter [Providing Access Based on Activities for Business Users \[page 819\]](#), you learn how to implement access for different activities, for example, separate read and write authorizations.
- In chapter [Providing Access Based on Field Values for Business Users \[page 829\]](#), you learn how to fine-tune your access restrictions depending on field values.

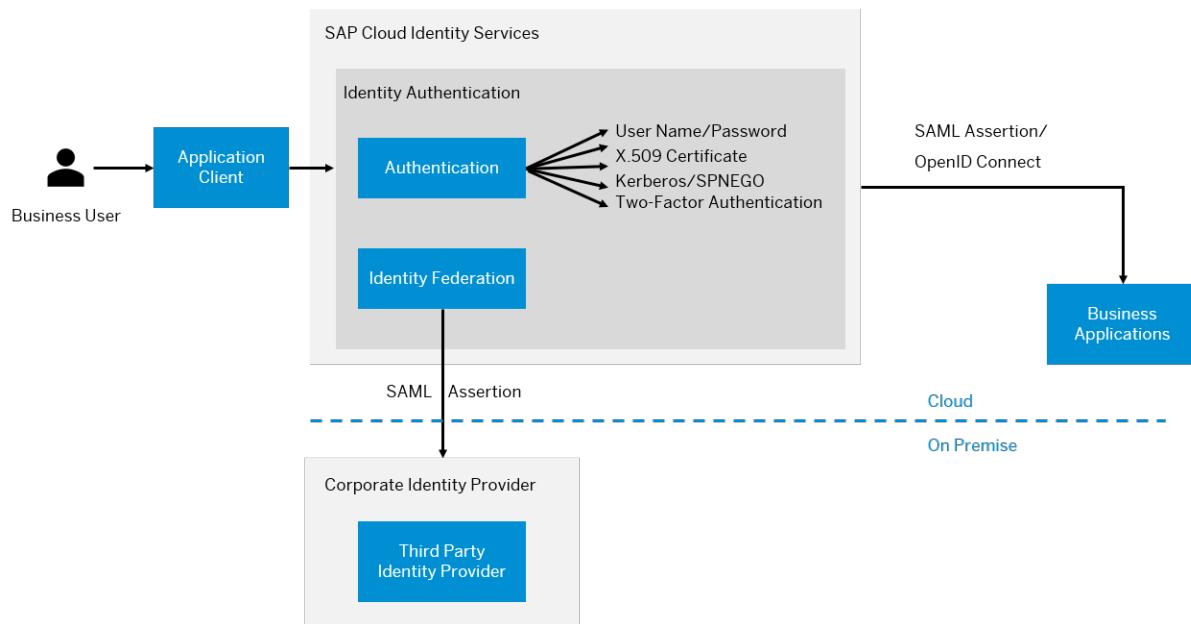
OData services that you develop using ADT can also be consumed in SAP Fiori applications (developed using the Business Application Studio). In this guide, you also learn how to provide access to an SAP Fiori application that consumes a business service.

## Custom Development and Testing

In SAP Business Technology Platform (SAP BTP), customers have the flexibility to develop their own code and services using ABAP. To test newly created services and applications in the development tenant, it is crucial that the developer role `SAP_A4C_BC_DEV_TST_PC` is extended with every newly created custom Authorization Object (AO), ensuring that full authorizations are granted for testing purposes. This ensures that developers can comprehensively test their applications without facing authorization issues, enhancing the development and deployment experience.

### 4.2.7.1 Identity Management

Identity management comprises setting up trust and providing credentials for business users.



When setting up your identity management architecture, the following uses cases are possible:

- Using the default identity provider SAP ID service, an instance of Identity Authentication service. It is used to set up trust for business users and supports identity federation, which enables user access across various systems by exchanging identity information based on a trust relationship. See [Identity Federation \[page 800\]](#).
- Establishing trust to a custom identity authentication service tenant. See [Setup of a Custom Identity Service \[page 202\]](#).
- Using a corporate identity provider in a hybrid landscape. In this case, we recommend using the Identity Authentication service as a proxy for the corporate identity provider. See [Corporate Identity Providers](#). Note that there is no direct trust relationship between the authenticating identity provider and the service provider that the business user is trying to access.

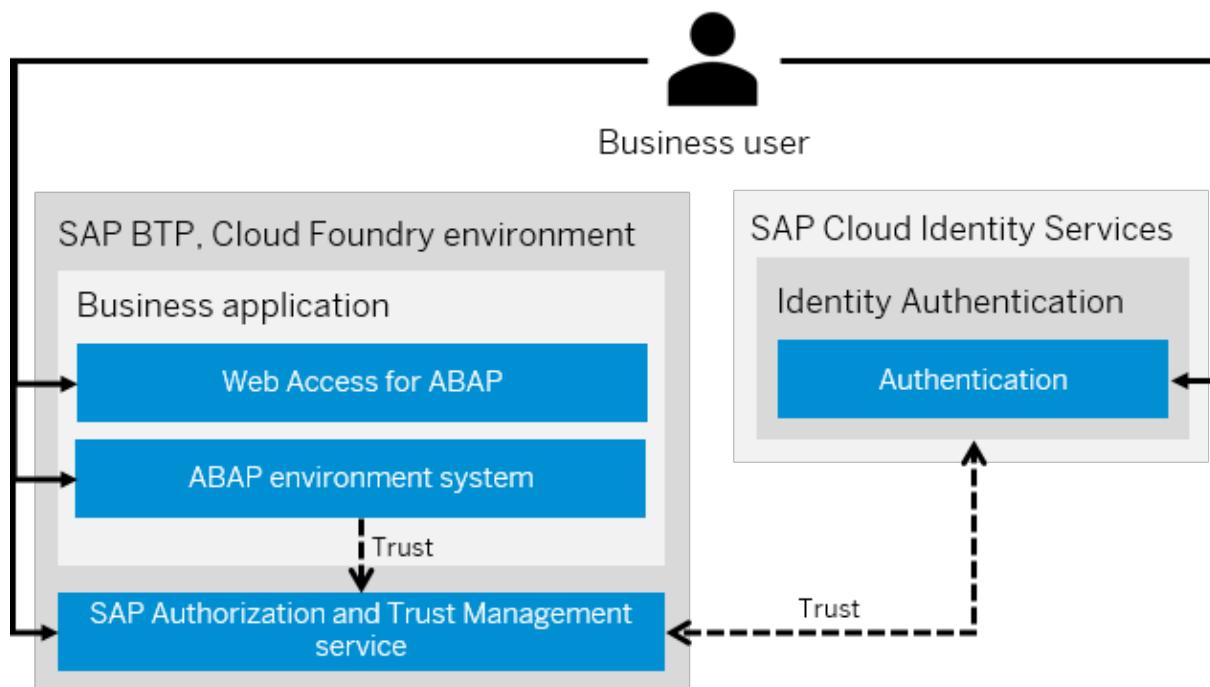
## Related Information

[Trust and Federation with Identity Providers](#)  
[SAP Cloud Identity Services - Identity Authentication](#)

### 4.2.7.1.1 Identity Federation

Identity federation is the process of sharing identity information between two parties by delegating the authentication responsibility to a trusted external party through the use of a common token.

In the ABAP environment, the authentication of business users is based on OAuth 2.0 authorization code flow, using either the [SAML 2.0](#) or [OpenID Connect](#) protocol and [JSON web token \(JWT\)](#) as a common access token.



The business application (OAuth 2.0 client) is on one hand represented by the Web Access for ABAP as the gateway component or a standalone approuter application deployed to the SAP BTP, Cloud Foundry environment. See [Subscribing to the Web Access for ABAP](#) and [Application Router](#).

On the other hand, an ABAP environment system is serving the business application resources (OAuth 2.0 resource server). See [Creating an ABAP System](#).

The SAP Authorization and Trust Management service in the SAP BTP, Cloud Foundry environment is providing endpoints to retrieve and exchange the authorization code for an access token (OAuth 2.0 authorization/authentication server). See [SAP Authorization and Trust Management Service](#).

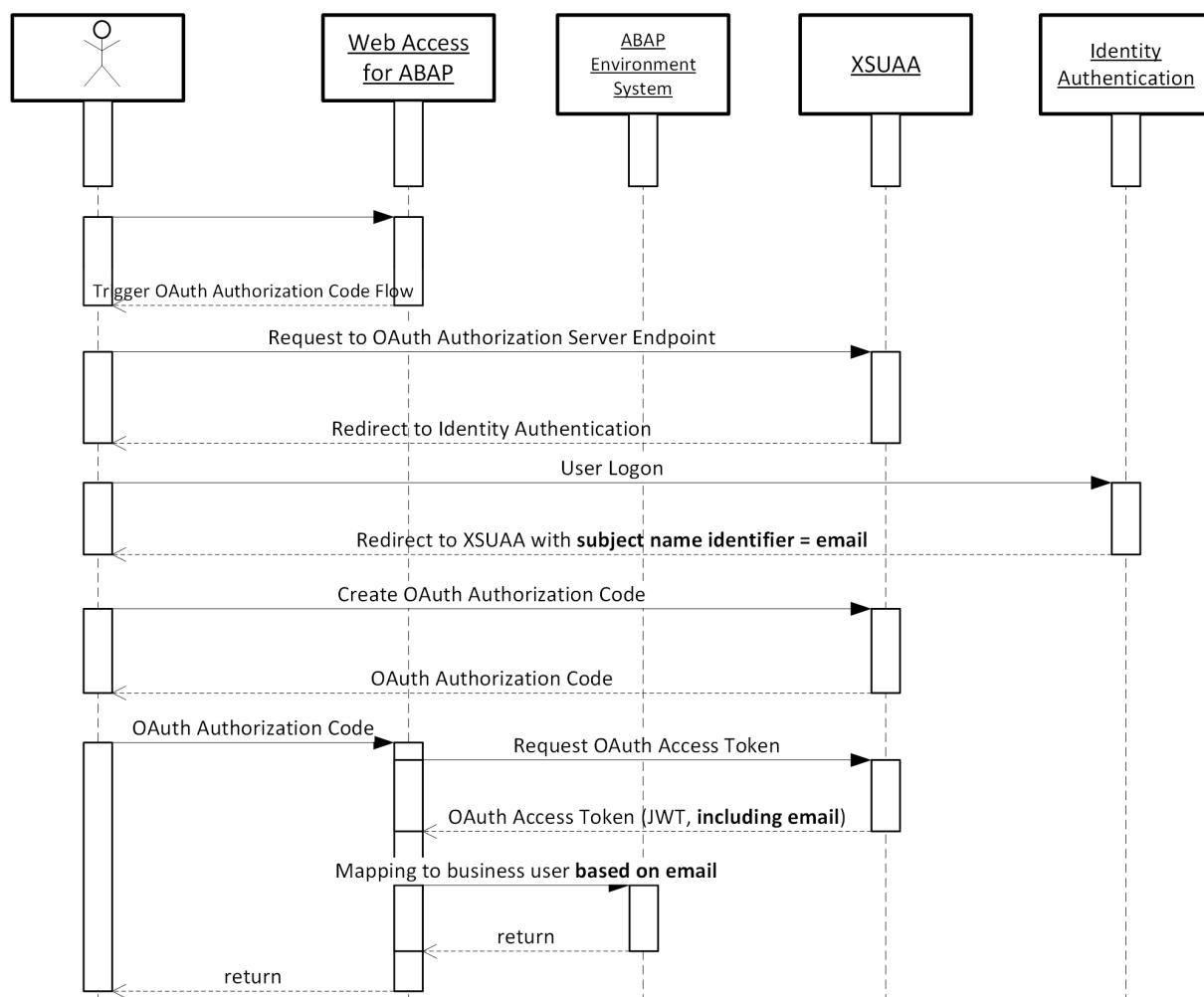
The Identity Authentication service is configured as an identity provider in the SAP BTP subaccount where the ABAP environment system has been created. The same subaccount is created as an application in the Identity Authentication service. See [Configuring Applications](#).

This configuration establishes a mutual trust relationship between SAP Authorization and Trust Management service (XSUAA) and the Identity Authentication service as identity provider. See [Trust and Federation with Identity Providers](#).

The ABAP environment system in turn trusts SAP Authorization and Trust Management service.

The business user is federated based on its e-mail address or user name, depending on the configuration of the login\_attribute during system provisioning (See [Creating ABAP System](#)) and the subject name identifier in the Identity Authentication application (see [Configure the Subject Name Identifier Sent to the Application](#)).

If a resource in the ABAP environment system is requested to be accessed by a business user, the steps to acquire the required access token are as follows:



1. The business user accesses the ABAP environment system through the Web Access for ABAP or a standalone approuter application. If no access token is available, the OAuth 2.0 authorization code flow is triggered.
2. A request to the authorization server endpoint in the SAP Authorization and Trust Management service (XSUAA) checks if the business user is already logged on. If the business user is not logged on, the business user is redirected to the Identity Authentication service as the identity provider used for logon.
3. After logging on, the response of the Identity Authentication service, including the e-mail address of the business user as the subject name identifier, is redirected to the authorization server endpoint of SAP Authorization and Trust Management service (XSUAA) to retrieve the authorization code.

The authorization code is necessary to retrieve the access token in the following steps. Finally, the authorization code is redirected again to the Web Access.

4. Based on the authorization code, a JSON web token (JWT) is retrieved from the access token endpoint in SAP Authorization and Trust Management service (XSUAA). This OAuth access token includes the e-mail address or login name of the business user that serves as a subject name identifier.
5. To consume resources, such as business services in the ABAP environment system, the JSON web token is forwarded as part of such requests and mapped internally to the business user.

**① Note**

Authentication of users other than communication users does not require identity federation as these technical users are authenticated locally in the ABAP environment system.

### 4.2.7.1.2 User Types

#### SAP BTP, ABAP environment Users

In the ABAP environment, there are different types of users such as business users, communication users, and support users.

	Business User	Communication User	SAP Support User	Provider Support User
<b>Owner</b>	Customer	Customer	SAP	Provider
<b>Purpose</b>	Accessing business applications (SAP Fiori Applications) and using ABAP Development Tools	System to system communication	Analyzing and troubleshooting	Analyzing and troubleshooting consumer tenants
<b>Authentication</b>	Identity Provider	Local	Local	Local
<b>Authorization</b>	Business Role	Communication Scenario Role	Support Role	Access to Identity and Access Management and Communication Management apps
<b>User ID</b>	CB*	CC*	_SAP*	PS*
<b>Access</b>	<host>.abap-web.<region>...<host>.abap-<region>...>...	<host>.abap-<region>...>...	<host>.abap-support.<region>...>...	<host>.abap-provider-web.<region>...>...

**Business users** are customer-owned end users that use SAP Fiori applications as well as applications and services in SAP BTP, such as SAP Business Application Studio. They are authorized by being assigned a specific business role. See [User Provisioning \[page 803\]](#).

### **ⓘ Note**

Developers using ABAP Development Tools are also considered business users.

**Communication users** are customer-owned technical users that are assigned to a communication system to enable integration with other solutions. For authentication, they can either be assigned a password or X.509 client certificate. See [Communication User \[page 884\]](#) and [Maintain Communication Users \[page 2593\]](#).

**Support users** are SAP-owned technical users that can access customer systems. See [SAP Support User Request Log \[page 2694\]](#).

**Provider support users** are technical users of the provider of a SaaS solution that can access consumer tenants in customer systems. See [Create Support Users](#).

## **SAP BTP Users**

In SAP BTP, there are platform and business users.

**Platform users** operate in the SAP BTP cockpit by deploying, administering, and troubleshooting applications and services. They are authorized by using role collections or by user assignment in the Cloud Foundry space as space or org members, such as space developers.

**Business users** use the applications that are deployed to SAP BTP and are authorized by using role collections.

To learn more about platform and business users in SAP BTP, see [User and Member Management](#).

## **Related Information**

[Business Catalogs for Identity and Access Management Apps \[page 2706\]](#)

### **4.2.7.1.3 User Provisioning**

User provisioning is the automated process of creating users, that are later granted permissions to access business services and applications, to allow system to system communication, or to perform troubleshooting.

### **ⓘ Note**

Please make sure that employee and business user data is maintained in accordance with the `login_attribute` system provisioning parameter (see [Creating ABAP System](#)) and the subject name identifier configured in the Identity Authentication application (see [Configure the Subject Name Identifier Sent to the Application](#)): If `login_attribute = email` is used, the email address at employee level needs to be maintained for identity federation to work, if `login_attribute = user_name` is configured, the username on business user level needs to match the login name provided in the identity authentication user.

## Creating SAP BTP, ABAP environment Users

Each business user created in the ABAP environment system has a unique and stable employee ID. Employees are maintained in the SAP Fiori app *Maintain Employees*. See [Maintain Employees \[page 2624\]](#).

### → Recommendation

We recommend setting the employee ID by an HR system, which allows you to update master data in the *Maintain Employees* app, for example, when a business user has a new email address because of a name change, or if you want to edit the user information of the initial admin of an ABAP environment service instance.

The following provisioning methods to automate identity lifecycle processes for business users are available:

- SAP Fiori app *Maintain Employees* with CSV file upload. See [Maintain Employees \[page 2624\]](#).
- SOAP service using communication scenario Identity Management Integration SAP\_COM\_0093. See [Inbound Service: Business User \[page 1048\]](#) and [Inbound Service: Business User - Read \[page 1065\]](#).
- Identity Provisioning service using communication scenario SAP Cloud Identity Services - Identity Provisioning SAP\_COM\_0193. See [Identity Provisioning - SAP BTP ABAP Environment as a Target System](#).

### → Recommendation

To create new business users in the ABAP environment, we recommend using the Identity Provisioning service.

This requires setting up the Identity Authentication service as a source system for identity provisioning and managing employee information in the user store. The Identity Provisioning service can then read this information and map the Identity Authentication logon name to the ABAP environment employee ID. See [Identity Authentication as a Source System](#) and [Tutorial: Provision Users into your SAP BTP ABAP Environment](#).

To create **communication users**, see [How to Create Communication Users \[page 2594\]](#).

SAP support users are owned and created by SAP. However, in the *Display Technical Users* SAP Fiori app, you can check when and why SAP support users accessed your customer system in the past 12 months. See [SAP Support User Request Log \[page 2694\]](#).

## Creating SAP BTP Users

To create **platform users** and **space members**, see [User and Member Management](#) and [Creating New Space Members and Assigning Space Developer Roles to Them](#).

## Related Information

[What Is Identity Provisioning?](#)

## 4.2.7.2 Access Management

Access management comprises the process of implementing and providing authorizations for your services.

### [Authorization Basics \[page 805\]](#)

If you haven't worked with an ABAP system before, you might find this introduction to the basic authorization concepts useful before you proceed.

### [Example: Authorizations for a Bonus Calculation Service \[page 811\]](#)

Let's assume that you have a bonus calculation service, which, in the example used here, is represented by an active service binding. You want to model authorizations for this service.

### [Providing Access to a Business Service for Business Users \[page 812\]](#)

For a newly created business service, you must define how business users can access it and which activities are allowed.

### [Providing Access to the SAP Fiori Application \[page 846\]](#)

Learn what developers and administrators need to do so that an SAP Fiori application appears in the SAP Fiori launchpad of an authorized business user.

### [Providing Access to a Business Service for Communication Users \[page 849\]](#)

For a newly created business service, you must define how communication users can access it and which activities are allowed.

### [Providing Read or Write Access to a Business Service Using Privileged Mode \[page 860\]](#)

With enablement of privileged mode for a business object, it's possible to consume the business object without the need for additional business authorizations.

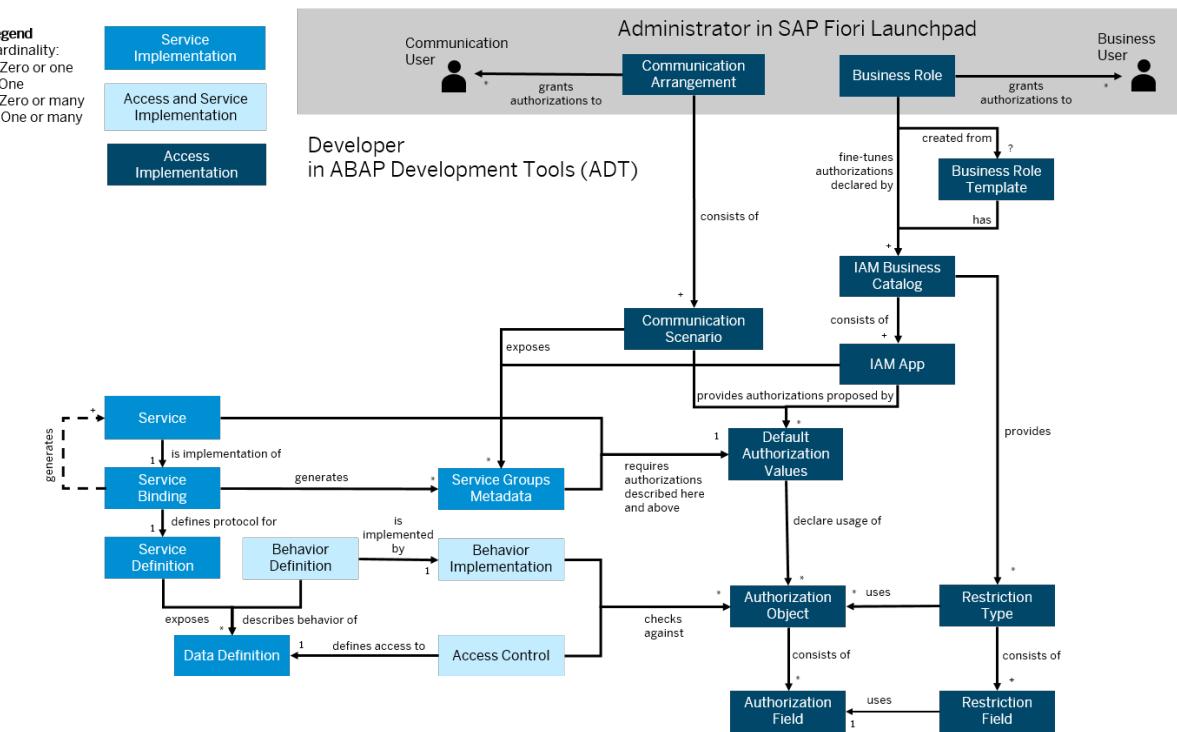
### [Troubleshooting \[page 862\]](#)

Learn more about apps and tools that help you identify and solve authorization issues.

## 4.2.7.2.1 Authorization Basics

If you haven't worked with an ABAP system before, you might find this introduction to the basic authorization concepts useful before you proceed.

Let's assume that you create business services. During the service implementation, you create a service definition, which exposes a data definition. In the ABAP RESTful Application Programming Model (RAP), you use behaviors to specify the operations and field properties of an individual business object. You can later enhance the behavior definition and implementation to include an authorization control. For more information about the ABAP RESTful Application Programming Model (RAP), see the documentation for the [ABAP RESTful Application Programming Model](#). The documentation of the ABAP RESTful Application Programming Model also describes the service implementation, that is, the creation of all objects highlighted in medium blue in the following graphic.



To provide access to your business service, you also create additional development objects in ABAP Development Tools (ADT). These additional objects help define which users can eventually access which data with their authorizations. Such objects can be authorization objects, authorization fields, IAM apps, IAM catalogs, and so on (highlighted in light and dark blue in the graphic above).

Let's have a look at each object.

## Authorization Field

An authorization field is the smallest unit in the area of authorizations. It's defined by a data element and its values; it's an attribute of a business object. An example of an authorization field is *Product Type*, with values such as *Material* or *Service* and the corresponding business object *Product*. In the example of a bonus management solution that is used in this guide, a possible authorization field is *Bonus Variant* with the values *Rate* and *Amount*.

A standard authorization field, which is already predefined in the system and which you can reuse to protect custom applications, is [Activity](#). Like every authorization field, [Activity](#) has multiple values, such as create, update, delete, release, and so on, for which you can check the user's authorization.

You can assign a check table to each authorization field. This table will serve as a value help when the field is consumed during the configuration of authorizations in the business role maintenance.

## Authorization Object

An authorization object groups up to 10 authorization fields that are related by AND. An authorization object ensures that complex authorization checks can run for multiple conditions. For example, you can create an

authorization object that defines which products users can access, such as products of type [Material](#), of type [Service](#), or both. Typically, you have the authorization field [Activity](#) in an authorization object because you want to define what kind of activities users can perform or not. Depending on the business context, you add more authorization fields, such as a purchase order type. In an authorization object, you also define which values of the [Activity](#) field you think are relevant.

After you have created authorization fields and authorization objects, you must implement checks against the authorization object to make sure that only authorized users can perform certain activities. During this check, the values specified in the program are compared with the values contained in the user's authorizations.

## Authorization Check

An authorization check determines whether the current user of a program has a certain authorization. The check compares programmed (fix or actual runtime) values that are required for a specific authorization field within a specific authorization object against the granted authorizations of the current user. As a developer, you can implement authorization checks in ABAP coding using the statement `AUTHORITY-CHECK` or using access controls.

## Access Control

ABAP Core Data Services (CDS) have their own concept for data protection and authorization: access controls. Access controls provide an easy way to implement protection against unauthorized read access based on field values.

For more information about access control, see [Access Controls](#) in the ABAP CDS Development User Guide.

## Authorization Control

Authorization control in the RESTful Application Programming Model protects your business object against unauthorized access to data. You can use global and instance-based authorization control. Global authorization is used for all authorization checks that only depend on the user. You can define global authorization to check if users are allowed to execute an operation in general. Instance authorization is used for all authorization checks that additionally depend on the state of the entity instance in question. With instance authorization, you can define authorization that depends on a field value of the instance.

Authorization control is part of the behavior implementation of a business service.

For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

## Default Authorization Values

You can define default authorization values for each authorization object and business service. These default authorization values are then automatically assigned when the business service is added to an IAM app or to a

communication scenario. Note, however, that the default authorization values can be overwritten by values set in the IAM app and by restrictions in the business role or communication scenario.

## Restriction Field

A restriction field is based on an authorization field. It's used to restrict the access to instances of a business object, for example, a product. Restrictions are only available when access to business users is provided.

With restriction fields, you can empower an administrator to create multiple business roles with different values in the restriction field. Without restriction fields, you, as a developer, need to create a separate IAM app and a business catalog for each business role.

For example, an administrator wants to create different business roles for product master specialists who maintain products in the system. With a restriction field that contains values for the different available product types, you can fine-tune business roles in such a way that some product master specialists only get the authorizations for maintaining products of type *Materials* and others also of type *Service*.

## Restriction Type

A restriction type is an authorization entity that bundles the available restriction fields according to a logical definition, for example, product type and product category. In a restriction type, you also define for which authorization object the restrictions are relevant. You can add restriction types to business catalogs.

For example, you can create a number of restriction fields that are relevant for creating business roles in product master data. You can, for example, create restriction fields such as *Product Type* and *Product Category*, and then bundle them using the restriction type *Product*.

## IAM App

An IAM app is a development artifact that you can use to define necessary authorizations for business users for one or more services that you created in ADT. You create the IAM app in ADT, add the services, and define the necessary authorizations needed for the services. After that, you can include the IAM app in a business catalog, which then allows you to include the IAM app and its related authorization definitions into a business role.

## IAM Business Catalog

In an IAM business catalog, you bundle multiple IAM apps and their predefined authorizations, for example, for a specific business area. The business catalog links what has been defined by development with settings done by administration. Business catalogs consist of IAM apps with their predefined authorizations, which can be reused in business roles. If you define an IAM business catalog with restriction types, you allow the administrator later to fine-tune the authorizations during business role maintenance.

You create an IAM business catalog in ADT. After you, as a developer, have published the business catalog in ADT, it's available in the SAP Fiori apps for business catalog and business role maintenance of the development system and can be used for user tests.

## Business Role Template

You can define business role templates to make it easier for administrators to find the business catalogs that are relevant for the corresponding role in your company.

For SAP Business Technology Platform, role templates for administrators and developers are available: The business role template `SAP_BR_DEVELOPER` includes permissions for using ABAP Development Tools (ADT) and for creating the development objects mentioned here. The business role template `SAP_BR_ADMINISTRATOR` includes permissions for creating business roles.

## Business Role

A business role is the administrative object that administrators use to group and fine-tune authorizations and assign them to business users. To create a business role, an administrator adds one or multiple IAM business catalogs to it. These IAM business catalogs contain predefined authorizations that allow users to access IAM apps. In business roles, administrators can set the authorizations for each access category, including available restrictions for different instances of a business object. If the IAM business catalog contains restrictions, administrators can also override authorizations for these restriction fields during business role maintenance.

## Access Categories

An access category defines what kind of access is granted to the business users assigned to a business role, for example, read, write, or value help.

Each activity from the corresponding standard authorization field is typed with an access category, for example, the activity *Display* has the access category *Read* and the activity *Create* has the access category *Write*.

The activities for which permissions are assigned using a business role are defined in the IAM apps. The activities that a business user is allowed are the result of restriction settings for each access category in a business role and the corresponding activities defined in the IAM apps in the business role.

If the business role doesn't give write access, the *Create* activity, for example, isn't allowed. While it's possible to block write access, it's not possible to forbid read access completely in a business role because a *No Access* option isn't available. Therefore, read access is either allowed or, with the use of restriction fields, it's restricted by the allowed values of the restriction fields. In this case, display is only allowed for business object instances matching the allowed restriction field values.

## Authorization Context

A CDS behavior definition can define authorization contexts that list multiple authorization objects. These authorization objects are used for the ABAP statement `AUTHORITY-CHECK OBJECT` in the implementation methods of the ABAP behavior pool of the business object.

The authorization contexts can be defined as follows:

- Own authorization context: Such an authorization context is defined using the statement `define own authorization context`.

### → Recommendation

We recommend that you define the own authorization context for all your custom business objects to document which authorization objects are checked in the logic of each business object.

- An authorization context with a specific name: Such an authorization context is defined using the statement `define authorization context`. With the authorization context with a specific name, you can define an authorization context that contains only a subset of the authorization objects from the own authorization context.

An authorization context can be useful, for example, to enable privileged mode for a business object (see [Providing Read or Write Access to a Business Service Using Privileged Mode \[page 860\]](#)).

For more information, see the ABAP keyword documentation that you can find in the ABAP Development Tools.

## Privileged Mode

With privileged mode in the behavior definition, RAP business object consumers can circumvent authorization checks, such as RAP authorization control or CDS access control. After enabling privileged mode, privileged read and write access to the business object is possible. In this case, the authorization checks for all listed authorization objects always return the value `authorized`.

For more information, see the ABAP keyword documentation in the ABAP Development Tools and [Authorization Control](#) in the guide to the ABAP RESTful Application Programming Model (RAP).

## Communication Scenario

A communication scenario in the ABAP environment is a development artifact that describes how two communication partners communicate with each other. It provides technical information, such as the used inbound and outbound services and their service type, for example OData or SOAP, and the number of allowed communication arrangement instances. It is also the place where you define the required authorizations that a communication user must have to use the services of the communication scenario.

For more information, see [Communication Scenario](#).

## Communication Arrangement

A communication arrangement in the ABAP environment is a runtime instance of a specific communication scenario. It grants the authorizations for the inbound and outbound services of the communication scenario to communication users. It grants the authorizations only from and to a specified partner, the so-called communication system. The administrator of the ABAP environment creates the communication arrangement in the SAP Fiori launchpad.

For more information, see [Communication Arrangement](#).

## Putting It All Together: Authorization

A user's authorization is influenced by the combination of permissible values in each authorization field of an authorization object. An authorization enables a user to perform an activity in the system, based on a set of authorization field values. However, the authorization object isn't the only entity in the system that influences what a user is allowed to do. IAM apps, communication scenarios, business roles, and restrictions in business roles can still change what was defined in the authorization object.

### 4.2.7.2.2 Example: Authorizations for a Bonus Calculation Service

Let's assume that you have a bonus calculation service, which, in the example used here, is represented by an active service binding. You want to model authorizations for this service.

#### The Bonus Calculation Service

The bonus calculation service is part of a sales bonus program where the sales revenues that sales reps were able to generate are used as the basis for their bonus plan. In the ABAP RESTful Application Programming Model (RAP), the bonus plan is modeled as a business object *Bonus Calculation* of type managed.

With the bonus calculation service, users, for example, managers in sales, can create, update, display, and delete bonus calculations as standard operations. As part of a bonus calculation, users enter an employee name, a bonus variant and validity period can be set, and the bonus be calculated, which is a non-standard, business object-specific activity. In the example used in this guide, the available bonus variants are based on *Rate* or *Amount*, that is, the bonus recipient either gets a bonus based on a rate or a fixed amount.

The service can be consumed either by business users via a user interface (UI) or by technical communication users as a Web API. For both consumption types access has to be provided at least for nondevelopment systems because this access isn't automatically available in such systems.

## **Consumption in a UI for Business Users**

If the business service is consumed by a UI, the available UI enables business users easily, depending on their authorizations, to set data or to execute the actions the service is offering. For example, a UI for managers lets them perform activities on bonus calculations. The UI wraps the underlying service calls for the business users into intuitive UI elements. Changes that the business user isn't authorized for can either be made unavailable on the UI or their corresponding service call is done and an appropriate error returned from the service. The UI can act as a kind of authorization filter. This kind of service consumption always requires business user action.

## **Consumption as API with a Communication User**

If the business service is consumed as API, no UI wraps the allowed changes to the business service. Instead, the service caller can try to change all data and execute all actions directly and always gets errors if there are missing authorizations to do it.

In the example used here, the bonus calculation is an inbound business service that business users use to create new bonus calculations. There's no dedicated bonus calculation in the HR system, but instead, the HR business service calls the bonus calculation business service in the ABAP environment to set data and execute actions.

Where you work with an API business service (inbound or outbound) from or into the ABAP environment, you need to provide authorizations to technical users, so-called communication users, instead of business users. This service consumption can be performed automatically without any user action.

### **4.2.7.2.3 Providing Access to a Business Service for Business Users**

For a newly created business service, you must define how business users can access it and which activities are allowed.

#### **Motivation**

If you have the developer role `SAP_BR_DEVELOPER`, for example, you can test your new service immediately without having to create an IAM app and assign it to a business catalog and a business role. Activating the service binding, for example, creates the IAM app and assigns it to the business catalog `SAP_CORE_BC_EXT_TST` that is shipped to customers for testing purposes. This business catalog is assigned to the developer role template `SAP_BR_DEVELOPER`. Therefore, when you create a new service in a development system and activate the service binding, all users with this developer role can automatically access and use the service. Of course, all users with another role containing this catalog are then also able to access and use the service.

This kind of access isn't available anymore when you pull the business service into nondevelopment systems, for example, where this test catalog feature isn't enabled.

For business users, you must provide access to a newly created service because access isn't automatically available. However, all activities are allowed as long as you don't protect your business service. Therefore, you must develop identity and access management artifacts, which is explained in detail in this guide. They ensure that you can define the necessary authorizations and enable business users to consume the business service.

## Overview of Development Options

Now, let's assume you would like to achieve that certain business users are able to view and edit bonus calculations:

- Under *Providing Unrestricted Access*, you learn how to provide users read and write access to all bonus calculations.  
This scenario is simple to implement. It's enough if you have one user group that is allowed to make full use of the business service while all other users don't need access, not even read access. In this case, if managers have access to the business service, they can create, update, display, and delete all available bonus calculations.
- Under *Providing Access Based on Activities*, you learn how to create two business roles for separate read and write authorizations.  
With this scenario, you can keep business users with read access only from changing bonus calculations.
- Under *Providing Access Based on Field Values*, you learn how to fine-tune your access restrictions to particular fields, that is, on the level of data elements, such as the bonus variant.  
This is the most complex scenario, but it allows administrators to define business roles that authorize users to view and change only particular data; other data is filtered. For the bonus calculation service, managers can only create, view, update, and delete bonus calculations that have a specific bonus variant. The advantage of this scenario is that you, as a developer, can delegate some decisions about the business role design to the administrator. The administrator can decide which data is visible to business users, based on restriction fields in the business role.

After you've implemented access based on one of the scenarios mentioned, you can then also go on and implement access to the corresponding SAP Fiori application for the business service.

## Tutorial

In addition to this documentation, the following tutorial is also available:

[Create Authorization Model with SAP BTP, ABAP Environment](#) 

### 4.2.7.2.3.1 Considerations for Implementing Access and Authorizations

You can implement data access and authorizations for a service differently, depending on how you want to implement differences on the UI for different target groups. It also matters how you want to design business roles. Therefore, before you start with the actual implementation of data access and authorizations, you need to consider how you want to proceed.

There are different points in application development where you can start separating data access for different user groups. As a result, you get different business roles that are assigned to business users.

## Different Services for Different User Groups

With this approach, you create different services that grant different data access, depending on the user group and its corresponding role. Here, you already consider the required data access for the various user groups when you implement the services. As a result, you also implement different UIs, authorization defaults, IAM apps, and business catalogs, at least one per service. That way, you invest more effort in service development, but less in UI development because it's already clear from the data exposed by the service what the UI will provide.

Since you create different services for the user groups, system administrators would then also assign different business roles to business users that authorize them for one of the services, depending on the user's tasks.

## Different UIs for Different User Groups

If different user groups require a different structure or layout, another option is to implement different UIs for different user groups. While you need to invest more time to implement many separate UIs, especially if you need many of them, the effort for each UI is lower in comparison to dynamic UI handling.

A typical use case would be to use service bindings on different projections for the same business object. For example, a business object *Leave Request* has *Create*, *Delete*, and *Approve* activities. There could be two projections: one for the employees that allows them to create and delete leave requests, whereas another projection is for the manager who approves them.

If you decide to create different UIs for different user groups, this means you need to create an IAM app for each UI. Each IAM app needs to be included into a separate business catalog, which in turn is then included into a separate business role. As a result, an administrator creates a different business role for each UI.

## Dynamic UI Handling

With dynamic UI handling, you implement a UI that changes dynamically, depending on the user group. For example, while one user group has full access to all data, another user group can only see data in display mode and actions are disabled, or data and possible actions aren't even shown. Dynamic UI handling is useful if there are only small UI differences for different user groups.

For example, you implement an app that calculates bonuses for sales representatives. With dynamic UI handling, you can define a business role for a user group where users can view all bonus calculations and create, change, and delete bonus variants. Other user groups can only display the bonus calculations (possibly only for one bonus variant), but buttons for creating, changing, or deleting bonus variants are grayed out.

The development of a dynamic UI can result in more effort because you need to model and implement authorization access, for example. However, you also save efforts because you only implement one UI for all user groups and thus you avoid redundancies that arise when you maintain multiple UIs for different user groups.

In the ABAP RESTful Application Programming Model (RAP), you realize dynamic UI handling using dynamic behavior implementation. For more information about dynamic behavior implementation, see [Implementing a Protection Against Unauthorized Write Activities \[page 837\]](#).

## Business Role Design

In addition to the choices you make for the implementation of different UIs for different user groups, you also need to consider how business roles need to be implemented for your services. Depending on the use case of your service or services, you want the administrator to create different business roles. It's crucial to understand that the possible design of business roles depends on the choices you make when you implement the service, its UI, and the authorizations.

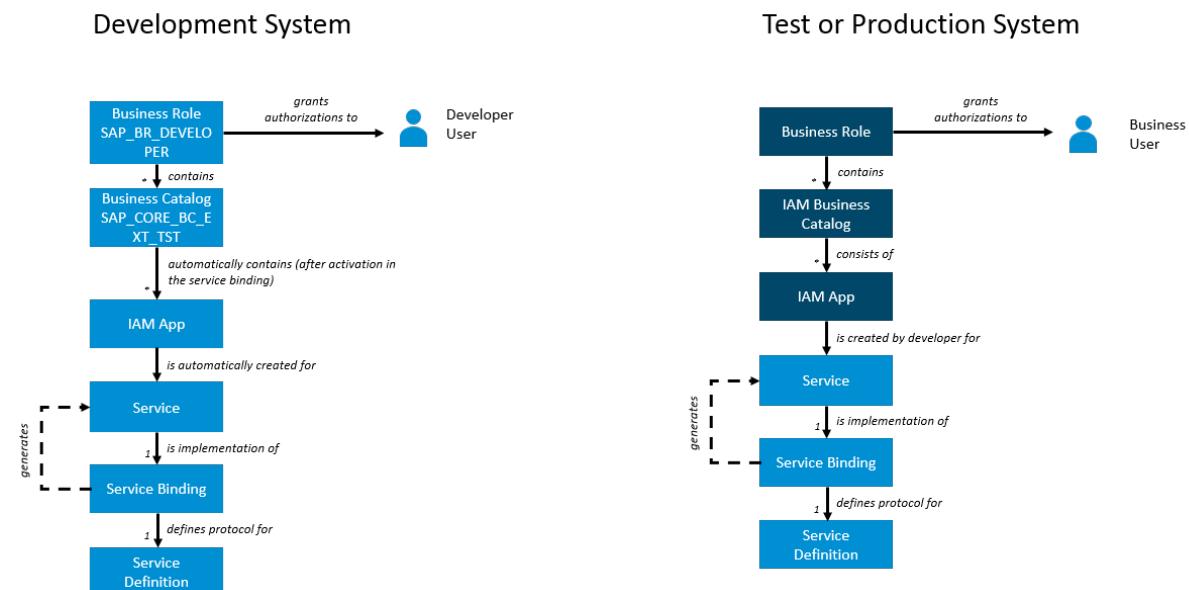
One possible approach is that you envisage business roles that can differentiate between the access categories of read, write, and value help (F4). This is possible using different business roles and doesn't require different authorization implementations.

The exception is when you need a finer distinction of activity authorizations, for example, between create/update and delete. In this case, you must implement different IAM apps for your service. On the basis of these IAM apps, you create different business catalogs, which the administrator uses to create different roles.

### 4.2.7.2.3.2 Providing (Unrestricted) Access for Business Users

In this scenario, you get information about how to provide unrestricted access with a few steps.

In this scenario, to provide access to the business service, you must create an IAM app and assign it to a business catalog. Administrators then create a business role that contains the business catalog and assigns the role to business users. Note that all objects that need to be created are highlighted in dark blue in the graphic.



This scenario requires only a few steps to provide authorizations, and it's the minimum that you must do to make your service available to business users. However, with this scenario, you always provide unrestricted write **and** read access. If you want to fine-tune the authorizations in more detail, check the scenarios *Providing Access Based on Activities* and *Providing Access Based on Field Values*.

## Related Information

[Providing Access Based on Activities for Business Users \[page 819\]](#)

[Providing Access Based on Field Values for Business Users \[page 829\]](#)

### 4.2.7.2.3.2.1 Developing Authorizations for Business Users (Developer)

For providing unrestricted access to a business service for business users, as a developer, you first create an IAM app, a business catalog, and, optionally, a business role template.

#### 4.2.7.2.3.2.1.1 Creating an IAM App for the Business Service

As a developer, you need to define an IAM app for the business service that you created in ABAP development tools (ADT). Creating an IAM app is the first step to be able to create a business role for your business service.

## Context

You create the IAM app in ADT. After the IAM app has been created, you can include the IAM app in a business catalog, which, in turn, can be included in a business role. This business role then can be used to control access to the app. For more information about IAM apps, see the [user guide](#) for ABAP development tools.

Instead of creating a new app, you can also use an existing IAM app for the service.

## Procedure

1. Follow the procedure in the user guide for ABAP development tools.

### ⓘ Note

In this scenario, there are no authorizations that you need to change on the [Authorization](#) tab page.

2. Save and publish the newly created IAM app locally.

## 4.2.7.2.3.2.1.2 Creating a Business Catalog

To be able to assign an IAM app to a business role, you first need to include the IAM app in a business catalog.

### Context

In a business catalog, you can bundle several IAM apps for which users should get authorized. You can assign a business catalog to a business role template.

In this simple scenario, no predefined, detailed authorizations are available for the IAM app, but you still need to create a business catalog. By including the IAM app in the business catalog, you simply create the prerequisites for allowing users a full access to your business service once they have the business role with the business catalog assigned.

Instead of creating a new business catalog, you can also reuse an existing business catalog.

#### → Recommendation

We recommend that you add the information to the business catalog description that the IAM app belongs to an unprotected service, so that this important information is available to administrators.

### Procedure

To create a business catalog, follow the instructions in the [user guide for ABAP development tools \(ADT\)](#).

Also follow the instructions to assign your IAM app to the business catalog.

## 4.2.7.2.3.2.1.3 Creating a Business Role Template (Optional)

In addition to a business catalog, you can also create a business role template with ABAP Development Tools (ADT). Creating a business role template is optional, but it can ease the business role creation for the administrator.

### Context

Creating a business role template especially makes sense if you have more than one business catalog to assign to a business role.

## Procedure

1. Create a business role template as described in [Creating a Business Role Template](#).
2. After the business role template has been created, add the business catalog to it.
3. Choose *Publish Locally* to make the business role template available for use in the development system.

### 4.2.7.2.3.2.2 Creating a Business Role and Assigning Business Users (Administrator)

As an administrator, to be able to grant authorizations to business users, you need to create a business role that you then assign to the business users.

## Context

Perform this task in every system. Alternatively, you can also create a business role once, download it, and upload it to other systems. For more information, see [How to Download and Upload Business Roles](#).

## Procedure

Follow the instructions of the documentation to create a business role from a template or from scratch (but omit the steps for maintaining instance-based restrictions and for managing launchpad space):

- [How to Create a Business Role from Scratch](#)
- [How to Create a Business Role from a Template](#)

#### ⚠ Caution

Even if you leave the write restrictions to their default *No Access*, business users still have write access with this business role in this scenario. Users always have full access because you haven't implemented an authorization check in the service behavior.

We recommend that you set the restrictions in the business role to unrestricted write authorizations to make it transparent to other administrators that this business role provides users with full read and write access.

## Results

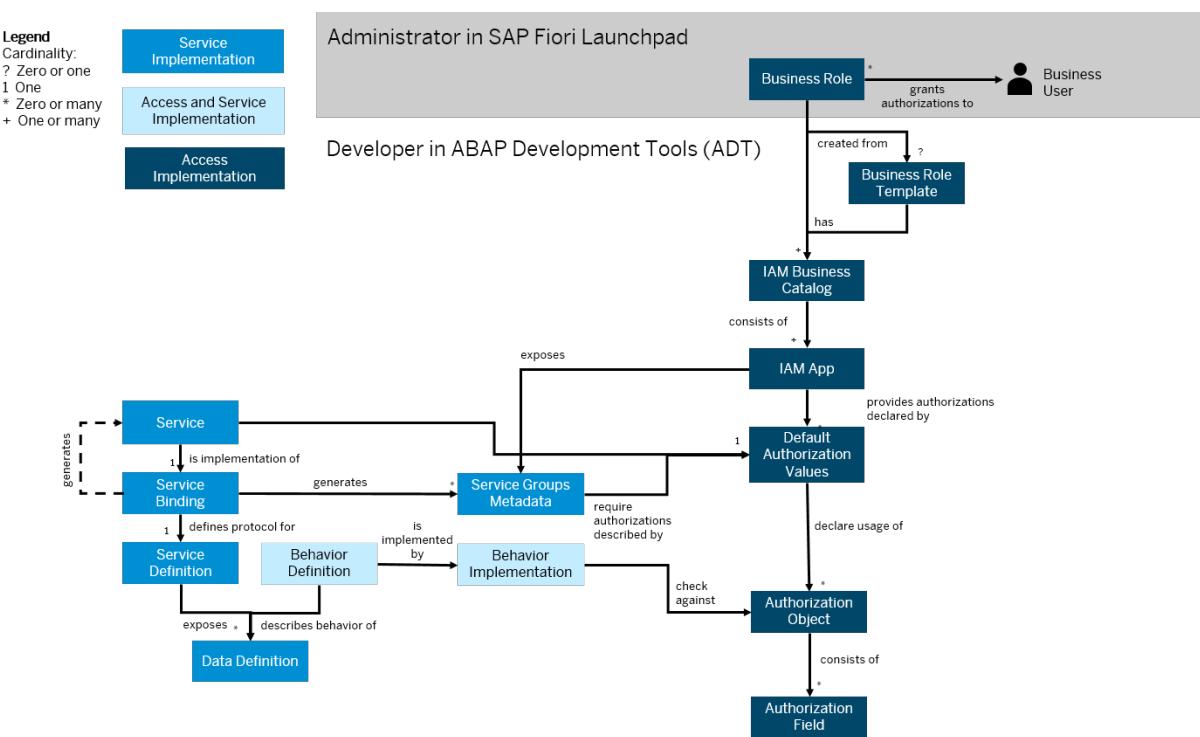
Users in the system have full read and write access to the service.

### 4.2.7.2.3.3 Providing Access Based on Activities for Business Users

In this scenario, you provide access depending on what the user should be allowed to do, for example, read or write access.

This scenario is more complex than a simple unrestricted access implementation because you must also create an authorization object and implement authority checks in the behavior implementation. However, you benefit from the extra efforts because you can enable the administrator to create separate business roles for different activities, that is, based on the authorization field ACTVT (*Activity*) for create, read, update, and delete.

## Overview



In the following, you get a more detailed overview of how to implement a separate read and write access. You can achieve such a separation by creating an authorization object with the standard authorization field ACTVT (*Activities*), an authority check to protect the service, an IAM app, and a business catalog. The administrator can then create two business roles for read and for write access each. If you want to enable the administrator to create more fine-tuned business roles, for example, to differentiate between the activities of create and delete, you proceed in a similar fashion, but you need to create two IAM apps (see also [Considerations for Implementing Access and Authorizations \[page 813\]](#)).

## Process

To create access protection and provide authorizations, you perform the following steps:

### Protecting the Service from Unauthorized Use

As a developer, you perform the following steps in ABAP Development Tools (ADT):

1. Create an authorization object with standard authorization field *Activity* (ACTVT) and permitted values 01,02,03,06.
2. Implement authority checks in the behavior implementation of the service to allow create, update, and delete activities only to users who have the authorizations of the authorization object.

As a result, users cannot create, update, or delete the relevant business objects anymore, for example, bonus calculations.

### Developing Authorizations for the Service

As a developer, you perform the following steps in ADT:

1. Add the required authorization objects to the list of default authorization values.
2. Create an IAM app and assign the service binding to it. Under *Authorizations*, select all activities (*01 Add or Create, 02 Change, 03 Display, 06 Delete*) to allow them for business users.
3. Create a business catalog and assign the IAM app to it.
4. Optional: Create a business role template and assign the catalog to it.

### Granting Authorizations to Business Users

As an administrator in the ABAP environment, you can now grant authorizations to business users for this service:

1. Create a business role. For business roles with write access, set unrestricted write authorization. For read-only business roles, keep the default, which is no access.
2. Assign the business role to the business users.

As a result, business users can view, create, update, or delete bonus calculations again, depending on their business roles.

### 4.2.7.2.3.3.1 Protecting the Service (Developer)

With creating an authorization object for your service and with the implementation of a protection against unauthorized creation and change activities, you implement a basic protection of your service.

After this, you can implement how authorizations are granted to business users.

## Related Information

[Developing Authorizations for Business Users \(Developer\) \[page 824\]](#)

## 4.2.7.2.3.3.1.1 Defining Permitted Activities in an Authorization Object

To define the permitted activities for a service, create an authorization object with the standard authorization field *Activity* (ACTVT) for the permitted activities.

### Procedure

To define the permitted activities, follow the procedure for defining authorization objects in the [user guide for ABAP development tools](#) (ADT).

#### ⓘ Note

In our example, in the newly created authorization object, under *Authorization Fields*, you only need the predefined *Activity* authorization field to define the permitted activities for the business users of your service. In our example, these activities would be *01 Add or Create*, *02 Change*, *03 Display*, and *06 Delete* for standard activities. In addition, you need *93 Calculate* for the business object-specific activity *Calculate Bonus*. If you click on an empty row and choose `ENTER`, the list of available values is shown.

In theory, by using the syntax `authorization: update` on an operation in the entity's behavior definition, you could use the authorization check that is implemented for the update operation for the annotated operation. This is also referred to as "authorization delegation". However, for security reasons, we recommend that you carefully consider whether you want to use authorization delegation because in this case, a user who is allowed to update a business object is then also allowed to perform all other activities on the business object.

### Results

In the following, the newly created authorization object for the bonus calculation example is called `ZBNNSCLC_AO`.

## 4.2.7.2.3.3.1.2 Modeling a Protection Against Unauthorized Write Activities

To protect your service from activities such as create, update or delete by unauthorized users, you can use the authorization controls that are available for services based on managed business objects.

### Context

In the example used here, the service is based on a managed business object *Bonus Calculation*. The use of a managed business object allows you to benefit from the global authorization for managed business objects,

which is part of the ABAP RESTful Application Programming model. For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

## Procedure

1. In ABAP development tools, call up the behavior definition of your service.
2. In the behavior definition, define authorization controls for your service.

In the bonus calculation example, they can be as follows:

### ↔ Sample Code

```
managed;
define own authorization context
{
  'ZBNNSCLC_AO';
}
define behavior for z_i_bonus_calculation alias calculation
implementation in class zbp_bonus_calculation unique
persistent table zbonusclc
lock master
// Identity and Access Management:
// Enable RAP managed authorization check at create, update, and delete
// -> requires implementation of two methods, one with the addition "FOR
INSTANCE AUTHORIZATION"
// and one with the addition "FOR GLOBAL AUTHORIZATION"
authorization master ( instance, global )
```

With the line `authorization master ( global )`, an authorization check for the standard operations `create`, `update`, and `delete` and for non-standard operations, for example, `calculate bonus`, is defined. You must implement the authorization check using a method with addition `FOR GLOBAL AUTHORIZATION` in the behavior implementation.

The sample code also contains the definition of your own authorization context. The own authorization context documents all authorization objects that are used by the business object implementation. In the bonus calculation example used in this documentation, this is the authorization object `ZBNNSCLC_AO` because it's checked in the authorization control. The own authorization context is also used to prefill authorization defaults.

### 4.2.7.2.3.3.1.3 Implementing a Protection Against Unauthorized Write Activities

To protect a service such as the example of bonus calculation against unauthorized bonus creations, updates, or deletions, enhance the behavior implementation of the service in ABAP development tools (ADT). The implementation is required by what has been defined in the behavior definition.

#### Context

Among the methods of the behavior implementation, you add methods that check the authorizations for creation and for update and delete.

#### Procedure

1. In the *Source Code Library* folder of your package, under *Classes*, choose the behavior implementation of your service.
2. In the `PRIVATE SECTION` in the handler class of the behavior implementation, define the methods for authorization checks for creation, update, and deletion.

##### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR <behavior definition or  
    its alias> RESULT result.
```

In the bonus calculation example, these method definitions could be the following:

##### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR calculation RESULT  
    result.
```

3. Add a method implementation that checks the authorizations for creations, updates, and deletions.

##### ⓘ Note

For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

In the bonus calculation example, this method could be the following:

##### ↔ Sample Code

```
METHOD get_global_authorizations.  
  
    IF requested_authorizations-%create EQ if_abap_behv=>mk-on.
```

```

*      check create authorization
AUTHORITY-CHECK OBJECT 'ZBNNSCLC_AO' ID 'ACTVT' FIELD '01'.
result-%create = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE
if_abap_behv=>auth-unauthorized ).
ENDIF.
IF requested_authorizations-%update EQ if_abap_behv=>mk-on.
*      check update authorization
AUTHORITY-CHECK OBJECT 'ZBNNSCLC_AO' ID 'ACTVT' FIELD '02'.
result-%update = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE
if_abap_behv=>auth-unauthorized ).
ENDIF.
IF requested_authorizations-%delete EQ if_abap_behv=>mk-on.
*      check delete authorization
AUTHORITY-CHECK OBJECT 'ZBNNSCLC_AO' ID 'ACTVT' FIELD '06'.
result-%delete = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE
if_abap_behv=>auth-unauthorized ).
ENDIF.
ENDMETHOD.
```

## Results

The implemented methods provide the desired protection from unauthorized uses:

- With the implementation of the method defined with the addition FOR GLOBAL AUTHORIZATION, an unauthorized business user gets an error message when trying to create a business object instance (in the example, a bonus calculation) or when trying to update or delete it via service call.
- Unauthorized business users can't use the *Delete* button in the list view on the service UI. In addition, the *Edit* and *Delete* buttons are hidden in the detail view on the service UI for a business object instance.

Now, the service is secured against unauthorized operations. As a next step, you must make sure that business users who need to use the service get the corresponding authorizations.

### 4.2.7.2.3.3.2 Developing Authorizations for Business Users (Developer)

After you've protected your service against unauthorized use, you can now create the objects that are needed to grant authorizations for business users: authorization default values, an IAM app, a business catalog, and a business role.

Follow the steps for editing authorization default values and for creating an IAM app, a business catalog, and a business role template. You should define own authorization context in the behavior definition first, which can be used to prefill authorization defaults.

#### *i* Note

For developing authorizations for communication users, the procedure is different. For more information, see [Developing Authorizations for Communication Users \(Developer\) \[page 857\]](#).

## 4.2.7.2.3.3.2.1 Editing Authorization Default Values

Authorization default values are automatically created when you create a service binding. You can add authorization objects and change the default authorization values.

### Context

The authorization default values are the authorizations that are assigned to business users by default if they have a business role based on the service. Note, however, that the authorization default values can be overwritten by values set in the IAM app. Therefore, if you want to avoid double maintenance, try and define the authorization default values in such a way that they can be used by all derived IAM apps.

For more information about authorization default values, see the [user guide for ABAP development tools](#).

### Procedure

1. Open the service binding and choose the *Maintain Authorization Default Values* link.
2. Choose *Synchronize* to add the latest authorization objects from the own context.

**ⓘ Note**

The *Synchronize* function only **adds** authorization objects that are not yet part of the list of authorization objects; it does not remove objects.

- 3. To check or change the authorization default values that are assigned automatically, select the authorization objects in the list.  
You can now specify what activities you want to authorize.
- 4. Choose the authorization object from the list and choose *Default With Field Values* from the dropdown list.  
In our example, we authorize for the standard activities *Create*, *Display*, *Change*, and *Delete* as well as the business object-specific activity *Calculate*.

### Results

Now, the standard authorizations for the service are defined and will be applied to an IAM app based on the service. You still can change them later in the IAM app itself.

### Related Information

[Defining an IAM App for the Business Service \[page 826\]](#)

## 4.2.7.2.3.3.2.2 Defining an IAM App for the Business Service

To assign a business user to a business role for your service, you need to create an IAM app, which can then be included into a business catalog, which, in turn, can be assigned to a business role.

### Context

Since you want to make your service available to users with a certain business role, you need to define an IAM app for the service that you created in ADT. The IAM app is only relevant for identity and access management (IAM), and you create the IAM app in ADT. You can use an IAM app to define authorizations for one or more business services.

Instead of creating a new IAM app, you can also use an existing IAM app for the service.

For more information, see the information about IAM apps in the [user guide for ABAP development tools](#) (ADT).

### Procedure

To create an IAM app, follow the instructions in the [user guide for ABAP development tools](#).

For the bonus calculation IAM app used as an example here, you have already specified before in the authorization default values what you want to authorize. If needed, you can overwrite these defaults again here.

#### ⓘ Note

In this example scenario, we assume that you create only one IAM app for access using all permitted activities and their access categories (write, read, value help). Be aware at this point that in the business role Fiori app, the administrator can later separate authorizations depending on the access categories (write, read, or value help). Therefore, there's no need to develop different types of access if you just want to separate write, read, and value help.

However, it's different if you want to have different business roles for activities within one access category, for example, one for the create and update activities and another for the delete activity, which are all write activities. In this case, you must start here to separate authorizations: Create separate IAM apps, one for each write activities group that should be separated, go on with separate business catalogs, and finally enable the administrator to create separate business roles based on the business catalogs.

### **4.2.7.2.3.3.2.3 Creating a Business Catalog**

To make predefined authorizations for your business service available to administrators to use, create a business catalog.

#### **Context**

In the business catalog, you can bundle multiple IAM apps and their predefined authorizations. You assign a business catalog to a business role. If you have more than one business catalog for the same business role, you might also consider creating a business role template.

Instead of creating a new business catalog, you can also reuse an existing business catalog for the app.

#### **Procedure**

To create a business catalog, follow the instructions of the ABAP development user guide: [Creating a Business Catalog](#).

Also follow the instructions to assign your IAM app to the business catalog.

### **4.2.7.2.3.3.2.4 Creating a Business Role Template (Optional)**

In addition to a business catalog, you can also create a business role template with ABAP Development Tools (ADT). Creating a business role template is optional, but it can ease the business role creation for the administrator.

#### **Context**

Creating a business role template especially makes sense if you have more than one business catalog to assign to a business role.

#### **Procedure**

1. Create a business role template as described in [Creating a Business Role Template](#).
2. After the business role template has been created, add the business catalog to it.
3. Choose [\*Publish Locally\*](#) to make the business role template available for use in the development system.

### **4.2.7.2.3.3.3 Creating a Business Role and Assigning Business Users (Administrator)**

To be able to grant authorizations to business users, you need to create a business role that you then assign to the users.

#### **Context**

You can create a business role from a business role template or from scratch.

Perform this task in every system. Alternatively, you can also create a business role once, download it, and upload it to other systems.

#### **Procedure**

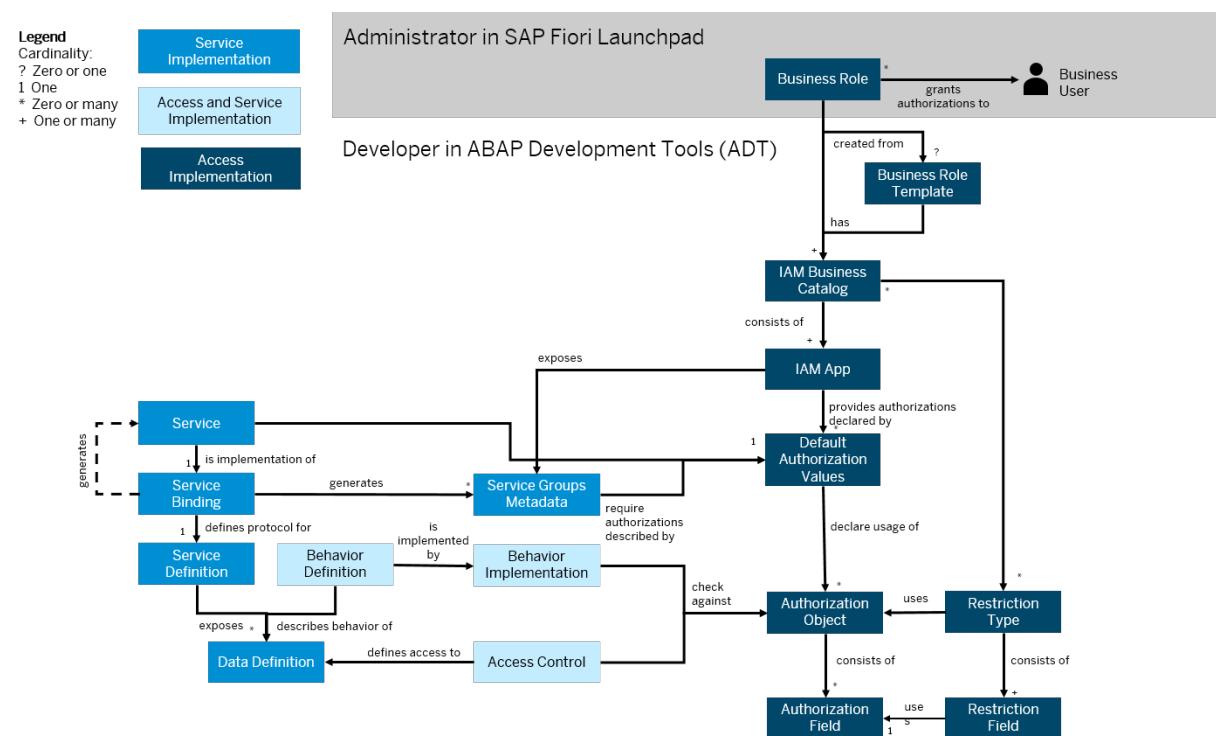
1. To create a business role, proceed as follows:
  - To create a business role from scratch and assign business users, follow the instructions in [How to Create a Business Role from Scratch](#).
  - To create a business role from a business role template and assign business users, follow the instructions in [How to Create a Business Role from a Template](#).
2. For roles with write access, set unrestricted write authorization. For read-only roles, leave the write restrictions as they are (the default is no access).

For more information about restrictions, see [How to Maintain Restrictions](#).

## 4.2.7.2.3.4 Providing Access Based on Field Values for Business Users

You can enable access to a service in such a way that it's dependent on the field values of a business object. As a result, business users can view or change only business object instances where the field values match their authorizations.

### Overview



The central concept in this scenario is the restriction field: In a restriction field, you can expose an existing authorization field as a restrictable field in a business role. You first create an authorization object and implement an access control and authority checks in the behavior implementation to protect your service from unauthorized use. Then you create an IAM app and a business catalog to make it available to administrators, who want to include the service (or, rather, the corresponding business catalog) into a business role. With the restriction field, you as a developer, can define which authorization fields should be available for the administrator during business role maintenance. Administrators can then decide which values of the restriction fields are relevant for some business roles. In the example of the bonus calculation service, administrators can create business roles to restrict the access for bonus calculations to selected bonus variants only.

## Process

To create access protection and provide authorizations, you perform the following steps:

### Protecting the Service from Unauthorized Use

As a developer, you perform the following steps in ABAP Development Tools (ADT):

1. Create an authorization field. For example, create an authorization field `ZBNS_VARN` for the bonus variant.
2. Create an authorization object with the standard authorization field `ACTVT` and other authorization fields if needed, for example, bonus variant.
3. Implement an access control as a read protection for the service.
4. Model authority checks in the behavior definition of the service.
5. Implement authority checks in the behavior implementation of the service to allow create, update, and delete activities only to users who have the authorizations of the authorization object.

As a result, no user can access the service to display bonus calculations or create, update, or delete bonus calculations anymore.

### Developing Authorizations for the Service

As a developer, you perform the following steps in ADT:

1. Add the required authorization objects to the list of default authorization values.
2. Create an IAM app and assign the service binding to it.
3. For each authorization field that you want to expose and consider in a business role, create a corresponding restriction field.
4. Assign the restriction fields to a restriction type.
5. Create a business catalog and assign the IAM app to it. Add the restriction types to the business catalog.
6. Optional: Create a business role template and assign the catalog to it.

Now, administrators can start granting access to business users. Development is finished and testing can start.

### Granting Authorizations to Business Users

As an administrator in the ABAP environment, you can now grant authorizations to business users for this service:

1. Create business roles and assign them to business users.  
For each business role, set the access category *Write* to restricted. Select the values for which you want to authorize the users. For example, create a business role with the authorization to create, update, and delete only bonus calculations that are based on specific bonus variants.

As a result, business users with the business role assigned can access the service to display bonus calculations or create, update, or delete bonus calculations again.

## 4.2.7.2.3.4.1 Protecting the Service (Developer)

With creating authorization fields and an authorization object for your service and with the implementation of a protection against unauthorized read and creation, you can implement a protection of your service against unauthorized use on field level.

After this, you can implement authorizations on field level that can be granted to business users.

### 4.2.7.2.3.4.1.1 Creating a Check Table for Authorization Fields

Create a check table that you can provide to the administrator as value help during business role maintenance.

#### Context

The advantage of having a check table is that the administrator has a value help when maintaining business roles with restricted access. As a developer, you provide access to a table where the field values are stored. When administrators create business roles and assign them to business users, they can call up the check table in the format of a value help and select the relevant values from the table.

In the bonus calculation example, the check table contains the bonus variants.

#### Procedure

1. In ABAP Development Tools, in the project explorer, right-click on the relevant package node.
2. In the context menu, choose *New* *Other ABAP Repository Object*.
3. In the following popup, choose *Core Data Services* *Data Definition* and *Next*.
4. Enter a name, description, and, as referenced object, enter the table where the values are stored that you want to provide for the administrator. In the bonus calculation example, this table is the table where the bonus variants are stored for the app. Choose *Next*.
5. Select a transport request and choose *Next*.
6. From the list of templates, choose *Define View* and choose *Finish*.
7. In the coding, choose the fields that you need to expose.

For the CDS view, it's sufficient that you use only those fields that are needed for a field help.

8. To provide some UI texts for the field help, add `@EndUserText .label` with a good label for each field that is exposed as part of the CDS view. The labels will be displayed as column headings in the field help of the business role maintenance.
9. Make sure that the annotation `@AbapCatalog.sqlViewName` is added at the beginning of the CDS view code.

For example, the code of such a CDS view can look as follows:

#### ↔ Sample Code

```
@AbapCatalog.sqlViewName: 'ZVARCHICK'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Check Table for bonus variants'
define view Z_VARIANTS_CHECK_TABLE as select from zbonusvar {
    @EndUserText.label: 'Bonus Variant'
    key variant as BonusVariant,
    @EndUserText.label: 'Category'
    category as Category,
    @EndUserText.label: 'Description'
    description as Description,
    @EndUserText.label: 'Created By'
    createdby as Createdby,
    @EndUserText.label: 'Created At'
    createdat as Createdat
}
```

10. Copy the SQL view name from `@AbapCatalog.sqlViewName` for later use (see [Defining an Authorization Field \[page 832\]](#)).

## 4.2.7.2.3.4.1.2 Defining an Authorization Field

As in the scenario for granting access based on activities, you define an authorization object. However, for this scenario, you first define authorization fields to fine-tune access on field level.

### Context

In the following, we assume that the bonus calculation service has a bonus variant field of type domain. You create an authorization field based on this data element.

### Procedure

1. To create an authorization field `ZBNS_VARN` for the bonus variant, following the procedure for defining authorization fields in the [user guide for ABAP development tools](#) (ADT).
2. In the *Data Element* field, enter the data element for the bonus variant.
3. In the *Check Table* field, enter the SQL view name that you have created as a check table for all possible bonus variants (see [Creating a Check Table for Authorization Fields \[page 831\]](#)).

#### ⓘ Note

You can use the [Search Help](#) button to simulate the field help as it's shown to the administrator during the business role maintenance.

### 4.2.7.2.3.4.1.3 Defining an Authorization Object

As in the scenario for granting access based on activities, you define an authorization object. However, for this scenario, you also add the authorization field ZBNS\_VARN for the bonus variant.

#### Procedure

Follow the instructions for defining authorization objects in the [user guide for ABAP development tools](#) (ADT).

#### → Tip

In our example, in the newly created authorization object, under [Authorization Fields](#), you need the following:

- You need the predefined [Activity \(ACTVT\)](#) authorization field to define the permitted activities for the business users of your business service: [01 Add or Create](#), [02 Change](#), [03 Display](#), and [06 Delete](#) as standard activities.
- If you have non-standard activities such as [Calculate Bonus](#) that you need to protect (and to permit later), check the list of existing values. If you click on an empty row and choose [ENTER](#), the list of available values is shown. For the bonus calculation example, let's choose [93 Calculate](#) for the business object-specific activity [Calculate Bonus](#).
- You need the newly created authorization field ZBNS\_VARN for the bonus variant.

### 4.2.7.2.3.4.1.4 Implementing a Read Protection

Prevent unauthorized read access to the data exposed by your service using access control. For this purpose, you create an access control and then implement it for the CDS view on which your service is based.

## Defining the Use of Access Control in the Data Definition

#### Context

For the data definitions that you want to protect, you need to define the use of access control.

### Note

If you're using additional data definitions of type projection view, you need to define the use of an access control in them.

## Procedure

1. Add the following line to the data definition of the interface:

```
@AccessControl.authorizationCheck: #CHECK
```

For more information, see [Access Control Annotations](#) in the documentation of the ABAP RESTful Application Programming Model.

2. If you're using additional data definitions of type projection view, you also need to define the use of an access control in them.

## Creating an Access Control

### Context

Even if you have modeled necessary authorizations for your service by creating a new authorization object, the authorization object itself doesn't protect your service from being accessed by a business user in the ABAP environment. Therefore, as a next step, you implement a read protection for your service. In our example, this means that business users should only be allowed to view bonus calculations whose bonus variant matches with the users' authorization. You implement a read protection using access control, which uses the authorization object that you created.

## Procedure

1. Go to the core data service (CDS) view on which your service is based.

You can find this CDS view under the data definitions of the core data services of your package.

2. To create an access control for the CDS view, follow the instructions of the ABAP CDS Development User Guide: [Creating Access Controls](#) and use the template [Define Role with PFCG Aspect](#).
3. If you're using additional data definitions of type projection view, you also need to create additional access controls for them using the template [Define Role with Inherited Conditions](#).

# Implementing the Access Control

## Context

You must edit the access control code to implement access control for the CDS view on which your service is based. Access control is based on the authorization object that you created before.

## Procedure

1. Choose the access control that you've created, which currently looks like the following:

### ↔ Sample Code

```
@EndUserText.label: '<your access control description>'  
@MappingRole: true  
define role <your access control name> {  
    grant  
        select  
        on  
            ${cds_entity}  
            where  
                (${entity_element_1}, ${entity_element_2}) =  
                aspect pfccg_auth(${authorization_object}, ${authorization_field_1}, $  
{authorization_field_2}, ${filter_field_1} = '${filter_value_1}');  
}
```

2. Adapt the code so that it grants access if the requesting business user has authorizations that contain authorization object ZBNNSCLC\_AO (*Bonus Calculation*) with allowed activity (ACTVT) *03 Display* and bonus variant (ZBNS\_VARNT) values that match the bonus variant (bonusvariant) of the existing bonus calculation:

### ↔ Sample Code

```
@EndUserText.label: 'Access Control for CDS Z_I_BONUS_CALCULATION'  
@MappingRole: true  
define role Z_I_BONUS_CALCULATION {  
    grant  
        select  
        on  
            z_i_bonus_calculation  
            where  
                (bonusvariant) = aspect pfccg_auth(ZBNNSCLC_AO,  
ZBNS_VARNT, ACTVT = '03');  
}
```

In our example, `cds_entity` is the `z_i_bonus_calculation` CDS view.

3. If you're using additional data definitions of type projection view, enhance the coding as follows:

### ↔ Sample Code

```
@EndUserText.label: '<your access control description>'  
@MappingRole: true
```

```

define role <your access control name> {
    grant
        select
        on
            <your projection view name>
                where
                    inheriting conditions from entity
z_i_bonus_calculation;
}

```

## Results

As a result, only business users with the corresponding authorizations get bonus calculations from the service, and they only get bonus calculations with allowed bonus variant values.

### 4.2.7.2.3.4.1.5 Modeling a Protection Against Unauthorized Write Activities

To protect your service from activities such as create, update or delete by unauthorized business users, you can use the authorization controls that are available for services based on managed business objects.

## Context

In the example used here, the service is based on a managed business object *Bonus Calculation*. The use of a managed business object allows you to benefit from the global and instance-based authorizations for managed business objects, which are part of the ABAP RESTful Application Programming model. For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

## Procedure

1. In ABAP development tools, call up the behavior definition of your service.
2. In the behavior definition, define authorization controls for your service.

In the bonus calculation example, they can look as follows:

#### ↔ Sample Code

```

managed;
define own authorization context
{
    'ZBNNSCLC_AO';
}
define behavior for z_i_bonus_calculation alias calculation

```

```

implementation in class zbp_bonus_calculation unique
persistent table zbonusclc
lock master
// Identity and Access Management:
// Enable RAP managed authorization check at create, update, and delete
// --> requires implementation of two methods, one with the addition "FOR
INSTANCE AUTHORIZATION"
// and one with the addition "FOR GLOBAL AUTHORIZATION"
authorization master ( instance, global )

```

With the line `authorization master ( instance, global )`, an authorization check for the standard operations `create`, `update`, and `delete` and for non-standard operations, for example, `calculate bonus`, is defined. You must implement the authorization check using methods with addition `FOR GLOBAL AUTHORIZATION` (for `create`) and `FOR INSTANCE AUTHORIZATION` (for `update`, `delete`) in the behavior implementation.

The sample code also contains the definition of your own authorization context. The own authorization context documents all authorization objects that are used by the business object implementation. In the bonus calculation example used in this documentation, this is the authorization object `ZBNSCLC_AO` because it's checked in the RAP authorization control and the CDS access control. The own authorization context can also be used to prefill authorization defaults.

#### 4.2.7.2.3.4.1.6 Implementing a Protection Against Unauthorized Write Activities

To protect a service, such as the example of bonus calculation from unauthorized creations, updates, or deletions, enhance the behavior implementation of the service in ABAP development tools (ADT). The implementation is required by what has been defined as authorization control in the behavior definition.

#### Context

Among the methods of the behavior implementation, you add methods that check the authorizations for creation and for update and delete.

For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

#### Procedure

1. In the [Source Code Library](#) folder of your package, under [Classes](#), choose the behavior implementation of your service.
2. In the `PRIVATE SECTION` in the handler class of the behavior implementation, define the method for authorization checks for update and deletion:

### ↔ Sample Code

```
METHODS get_instance_authorizations FOR INSTANCE AUTHORIZATION  
    IMPORTING it_entity_keys REQUEST requested_authorizations FOR  
<behavior definition or its alias> RESULT result.
```

In the bonus calculation example, this method definition can look as follows:

### ↔ Sample Code

```
METHODS get_instance_authorizations FOR INSTANCE AUTHORIZATION  
    IMPORTING keys REQUEST requested_authorizations FOR calculation RESULT  
result.
```

3. In the PRIVATE SECTION in the handler class of the behavior implementation, also define the method for authorization checks for creation:

### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR <behavior definition or  
its alias> RESULT result.
```

In the bonus calculation example, this method definition can look as follows:

### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR calculation RESULT  
result.
```

4. Add a method implementation that checks the authorizations for updates and deletions.

In the bonus calculation example, this method can look like this:

### ↔ Sample Code

```
METHOD get_instance_authorizations.  
*     As import table contains only keys, we need to get complete bonus  
calculation data for bonus variant value  
    READ ENTITIES OF z_i_bonus_calculation IN LOCAL MODE  
        ENTITY calculation  
        ALL FIELDS  
        WITH CORRESPONDING #( it_bonus_calc_keys )  
        RESULT DATA(lt_bonus_calcs).  
*     fill result list with authorizations per bonus calculation  
    LOOP AT lt_bonus_calcs INTO DATA(ls_bonus_calc).  
*         check update authorization for bonus calculation, incl. bonus  
variant dependency  
        AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '02' ID  
'ZBNS_VARNT' FIELD ls_bonus_calc-bonusvariant.  
*             set variable for update authorization  
        DATA(lv_update_allowed) = COND #( WHEN sy-subrc = 0 THEN  
if_abap_behv=>auth-allowed ELSE if_abap_behv=>auth-unauthorized ).  
*             check calculate bonus authorization for bonus calculation, incl.  
bonus variant dependency  
        AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '93' ID  
'ZBNS_VARNT' FIELD ls_bonus_calc-bonusvariant.  
*             set variable for calculate bonus authorization
```

```

DATA(lv_calculate_bonus_allowed) = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE if_abap_behv=>auth-unauthorized ).
*      check delete authorization for bonus calculation, incl. bonus
variant dependency
AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '06' ID
'ZBNS_VARNT' FIELD ls_bonus_calc-bonusvariant.
*      set variable for delete authorization
DATA(lv_delete_allowed) = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE if_abap_behv=>auth-unauthorized ).
*      fill result list
APPEND VALUE #( id = ls_bonus_calc-id %update = lv_update_allowed
%delete = lv_delete_allowed ) TO result.
ENDLOOP.
ENDMETHOD.
```

- Add a method that checks the authorizations for creation.

In the bonus calculation example, such a method could look like this:

#### ↔ Sample Code

```

METHOD get_global_authorizations.

IF requested_authorizations-%create EQ if_abap_behv=>mk-on.
*      check create authorization
AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '01'.
result-%create = COND #( WHEN sy-subrc = 0 THEN
if_abap_behv=>auth-allowed ELSE
if_abap_behv=>auth-unauthorized ).
ENDIF.
ENDMETHOD.
```

## Results

The implemented methods provide the desired protection from unauthorized uses:

- With the implementation of the method `get_instance_authorizations`, an unauthorized business user gets an error message when trying to update or delete a business object instance (in the example, a bonus calculation) via service call.
- With the implementation of the method `get_global_authorizations`, an unauthorized business user gets an error message when trying to save the creation of a new business object instance.
- Unauthorized business users can't use the `Delete` button in the list view on the service UI. In addition, the `Edit` and `Delete` buttons are hidden in the detail view on the service UI for a business object instance.

Now, the service is secured against unauthorized operations. As a next step, you must make sure that business users who need to use service get the corresponding authorizations.

## 4.2.7.2.3.4.2 Developing Authorizations (Developer)

After you've protected your service against unauthorized use, you create the objects that are needed for granting authorizations to business users to use your service.

For granting authorizations, follow the steps for restriction fields and types and for creating an IAM app, a business catalog, and a business role template. You should define your own authorization context in the behavior definition first, which can be used to prefill authorization defaults.

### 4.2.7.2.3.4.2.1 Editing Authorization Default Values

Authorization default values are automatically created when you create a service binding. You can add authorization objects and change the authorization default values.

#### Context

The authorization default values are the authorizations that are assigned to business users by default if they have a business role based on the service. Note, however, that the authorization default values can be overwritten by values set in the IAM app or by restrictions in the business role. Therefore, if you want to avoid double maintenance, try and define the authorization default values in such a way that they can be used by all derived IAM apps.

For more information about authorization default values, see the [user guide for ABAP development tools \(ADT\)](#).

#### Procedure

1. Open the service binding and choose the [Maintain Authorization Default Values](#) link.
2. Choose [Synchronize](#) to add the latest authorization objects from the own context.

##### ⓘ Note

The [Synchronize](#) function only **adds** authorization objects that are not yet part of the list of authorization objects; it does not remove objects.

3. To check or change the authorization default values that are assigned automatically, select the authorization objects in the list.

An authorization object that is used in the read-protecting access control automatically gets the settings [Default with Field Values](#) and the required activity value for [Display](#). Other authorization objects get [Default Without Field Values](#).

4. For the bonus calculation, you can now specify what you want to authorize, which are, in the example used here, the standard activities [Create](#), [Change](#), and [Delete](#) as well as the business object-specific activity [Calculate](#).

## Results

Now, the standard authorizations for the service are defined and will be applied to an IAM app based on the service. You still can change them later in the IAM app itself or delegate the value definition to the administrator during business role maintenance (only if restrictions are used).

## Related Information

[Defining an IAM App for the Business Service \[page 826\]](#)

[Creating Restriction Fields Based on Authorization Fields \[page 842\]](#)

### 4.2.7.2.3.4.2.2 Defining an IAM App for the Business Service

To assign a business user to a business role for your service, you need to create an IAM app, which can then be included into a business catalog, which, in turn, can be assigned to a business role.

## Context

Since you want to make your service available to users with a certain business role, you need to define an IAM app for the service that you created in ADT. The IAM app is only relevant for identity and access management (IAM), and you create the IAM app in ADT. You can use an IAM app to define authorizations for one or more business services.

Instead of creating a new IAM app, you can also use an existing IAM app for the service.

For more information, see the information about IAM apps in the [user guide for ABAP development tools \(ADT\)](#).

## Procedure

To create an IAM app, follow the instructions in the user guide for ABAP development tools.

For the bonus calculation IAM app used as an example here, you have already specified before in the authorization default values what you want to authorize. If needed, you can overwrite these defaults again here.

### ⓘ Note

In this example scenario, we assume that you create only one IAM app for access using all permitted activities and their access categories (write, read, value help). Be aware at this point that in the business role Fiori app, the administrator can later separate authorizations depending on the access categories

(write, read, or value help). Therefore, there's no need to develop different types of access if you just want to separate write, read, and value help.

However, it's different if you want to have different business roles for activities within one access category, for example, one for the create and update activities and another for the delete activity, which are all write activities. In this case, you must start here to separate authorizations: Create separate IAM apps, one for each write activities group that should be separated, go on with separate business catalogs, and finally enable the administrator to create separate business roles based on the business catalogs.

### 4.2.7.2.3.4.2.3 Creating Restriction Fields Based on Authorization Fields

For each authorization field that you want to expose and consider in a business role, you must create a corresponding restriction field and assign it to a restriction type.

#### Context

In the example used here, create a restriction field for the bonus variant field.

#### Procedure

For the procedure, see the ABAP development user guide: [Defining Restriction Fields](#).

### 4.2.7.2.3.4.2.4 Creating Restriction Types

Create a restriction type containing a restriction field and authorization object.

#### Context

Restriction types bundle restriction fields. In the example used here, there's one restriction type for the bonus calculation.

## Procedure

For the procedure, see the ABAP development user guide: [Defining Restriction Types](#).

### 4.2.7.2.3.4.2.5 Creating a Business Catalog with Restriction Types

Create a business catalog and add the restriction types to it.

## Context

In the business catalog, you can bundle multiple IAM apps and their predefined authorizations. You assign a business catalog to a business role. If you have more than one business catalog for the same business role, you might also consider creating a business role template.

In the business catalog, you can also define for which fields and access categories an administrator can maintain restriction values during business role maintenance.

Instead of creating a new business catalog, you can also reuse an existing business catalog for the app.

## Procedure

1. Create a business catalog and assign the IAM app to it.

For more information about creating business catalogs, see [Creating a Business Catalog](#) in the ABAP development user guide.

2. On the [Restriction Type](#) tab, enter the restriction type that you created.

In our example, add the restriction type [Bonus Calculation](#).

3. For each field, choose the appropriate checkboxes for write access, read access, and value help (F4).

#### Note

Even if you defined fewer values for the authorization field on which the restriction field is based in the IAM app (for example, one bonus variant only), the checkboxes here enable the administrator to allow all bonus variants again during business role maintenance.

4. Save the business catalog and publish it locally.

## 4.2.7.2.3.4.2.6 Creating a Business Role Template (Optional)

In addition to a business catalog, you can also create a business role template with ABAP Development Tools (ADT). Creating a business role template is optional, but it can ease the business role creation for the administrator.

### Context

Creating a business role template especially makes sense if you have more than one business catalog to assign to a business role.

### Procedure

1. Create a business role template as described in [Creating a Business Role Template](#).
2. After the business role template has been created, add the business catalog to it.
3. Choose *Publish Locally* to make the business role template available for use in the development system.

## 4.2.7.2.3.4.3 Granting Authorizations to Business Users (Administrator)

As an administrator, create business roles to grant authorizations to business users.

After a business catalog and optionally a business role template have been created by the developer, administrators take over. They grant authorizations to business users by creating and assigning business roles. The business role needs to contain the business catalog created by the developer to get all the predefined authorizations needed for the business service. If a business role template is available, the administrator can also use the business role template to create a new business role.

## 4.2.7.2.3.4.3.1 Creating a Business Role with Restrictions and Assigning It to Users

You can create a new business role where you can benefit from the predefined values and restrictions that were defined before.

### Context

The default is always no write access and unrestricted read access. You can have dedicated business roles that use these default settings. In the bonus calculation example, you can create multiple roles for changing only bonus calculation instances of a bonus variant.

For more information about maintaining restrictions, see [How to Maintain Restrictions](#).

Perform this task in every system. Alternatively, you can also create a business role once, download it, and upload it to other systems.

### Procedure

1. Create a business role from scratch or from a business role template, add the business catalog with the restrictions that the developer has created for the service.
2. Choose [Maintain Restrictions](#).
3. On the [Write, Read, Value Help](#) tab page, choose [Restricted](#).

The available restriction types and their fields are now shown, which is, in our example, the bonus calculation and its bonus variant.

4. Choose the edit button next to the restriction type.

In the following popup, the list of available values from the restriction field is shown.

5. Select the values for which you want to authorize the users.

In our example, the value would be one or multiple bonus variants that have been saved to the check table that the developer has assigned to the authorization field for the bonus variant (see [Defining an Authorization Field \[page 832\]](#)). As a result, business users with the business role assigned can create, update, and delete bonus calculations with the selected bonus variants only.

6. On the [Read, Value Help](#) tab page, choose [Unrestricted](#).

This means that business users with this business role can still view all bonus calculations independently of their bonus variant, even if they can only change bonus calculations with a particular bonus variant.

7. On the [Assigned Business Users](#) tab, assign the business users to your new business role

These business users get the authorizations as defined in the business role.

8. Save the business role to activate it.

## 4.2.7.2.4 Providing Access to the SAP Fiori Application

Learn what developers and administrators need to do so that an SAP Fiori application appears in the SAP Fiori launchpad of an authorized business user.

### Overview

For business services created in ABAP Development Tools, developers can also create SAP Fiori application UIs. Let's assume that they created UIs using, for example, SAP Business Application Studio or Visual Studio Code and deployed them back to the ABAP environment. To ensure that business users have access to the SAP Fiori application in their SAP Fiori launchpad, developers and administrators need to perform a few steps.

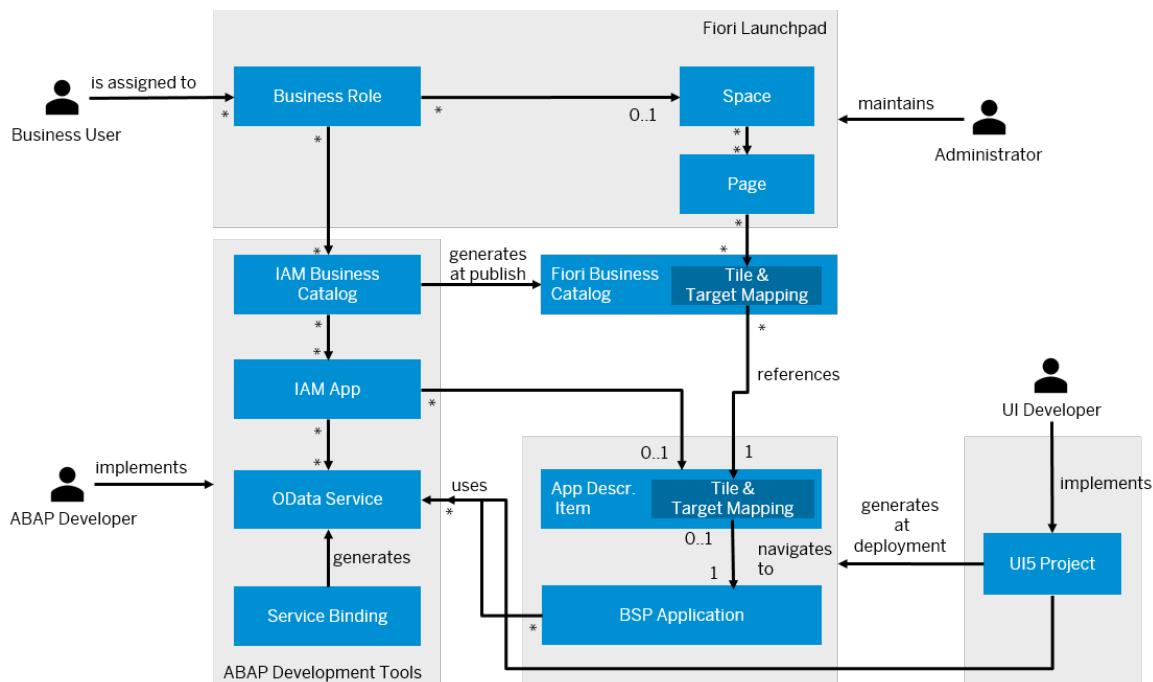
When you deploy an SAP Fiori application to the ABAP environment, the following objects, among others, are created automatically:

- A business server page (BSP) application, which stores the code of the SAP Fiori application
  - An SAP Fiori launchpad app descriptor item, which contains the tile and navigation target for the SAP Fiori launchpad integration
- The SAP Fiori launchpad app descriptor item is created from the `manifest.json` file of the UI5 app, where the tile and navigation parameters are configured.

You can find the BSP application and the SAP Fiori launchpad app descriptor item as objects in your deployment package in ABAP Development Tools.

After an SAP Fiori application has been deployed, it can't be immediately consumed in the SAP Fiori launchpad. You have already created an IAM app and added the business service for which you've created your UI to the IAM app. You must now also assign the automatically created SAP Fiori launchpad app descriptor item to this IAM app.

As an administrator, you must create and enable an SAP Fiori launchpad space and page for business users.



This guide outlines the steps that are needed after you've followed the procedures outlined in one of the scenarios *Providing (Unrestricted) Access*, *Providing Access Based on Activities*, or *Providing Access Based on Field Values*. For a general overview of all steps for developing an SAP Fiori application UI in the ABAP environment, see [Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#).

## Tutorial

In addition to this documentation, the following tutorial is available:

[Integrate List Report into ABAP Fiori Launchpad](#)

### 4.2.7.2.4.1 Assign the SAP Fiori Application to the IAM App (Developer)

To ensure consistent authorizations for the business service and its UI, you need to assign the SAP Fiori launchpad app descriptor item and the business service to the same IAM app.

## Context

After an SAP Fiori application has been deployed, it can't be immediately consumed in the SAP Fiori launchpad. You have already created an IAM app and added your business service to it. You must now also assign the SAP Fiori launchpad app descriptor item, which was automatically created during deployment, to this IAM app.

You can find the SAP Fiori launchpad app descriptor item as an object in ABAP Development Tools in the package to which you deployed the SAP Fiori application.

## Procedure

1. In ABAP Development Tools, open your IAM app.
2. In the *UI5 Application ID* field, enter the technical name of the SAP Fiori launchpad app descriptor item or use the field value help.
3. Save and publish locally.

## Results

If you haven't already done so, assign the IAM app to an IAM business catalog, and publish the IAM business catalog in ABAP Development Tools. Publishing generates a corresponding SAP Fiori business catalog that is listed in the SAP Fiori launchpad app *Business Catalogs*.

### 4.2.7.2.4.2 Integrating the SAP Fiori Application into the SAP Fiori Launchpad (Administrator)

To ensure that an SAP Fiori application appears in the SAP Fiori launchpad of the business users, you, as an administrator, need to perform a few configuration steps.

#### Prerequisites

You, as an administrator, have already created one or multiple business roles and assigned them to business users.

#### Context

You must integrate the SAP Fiori app into the IAM structure of the SAP Fiori launchpad of SAP Business Technology Platform.

#### Procedure

1. Create a space for the business role.
2. Add a page to this space and add the app tile to the page.

For more information about how to proceed, see [Create a New Space and Page for a Business Role](#) in the SAP Fiori launchpad documentation on SAP Help Portal.

## Results

You've now enhanced the business roles for the business service with authorizations for the UI. All users that have been assigned to the business roles before can now view the page and SAP Fiori application in their SAP Fiori launchpad.

### Note

The SAP Fiori application only appears in the spaces mode for the user. The spaces mode was developed to offer more flexibility to influence the launchpad layout for specific user groups. The business role defines which users see a specific space. For more information, see [Managing Launchpad Spaces and Pages](#) on SAP Help Portal.

## 4.2.7.2.5 Providing Access to a Business Service for Communication Users

For a newly created business service, you must define how communication users can access it and which activities are allowed.

Instead of accessing a business service using a business user, it is also possible to access a business service using a communication user. Communication users are relevant in scenarios where you work with an API business service (inbound or outbound) from or into the ABAP environment.

For communication users, you must provide access to a newly created service because access isn't automatically available. However, on instance level, all activities are allowed as long as you don't protect your business service. Therefore, you must develop identity and access management artifacts.

The main difference to providing access to a business user is that, as a developer, you do not create an IAM app and a business catalog but a communication scenario. Authorizations are maintained in the communication scenario. Administrators cannot restrict the authorizations defined in the communication scenario later on when they create communication arrangements based on the communication scenario. Therefore, you as a developer decide which authorizations are granted for the communication user by the administrator.

### Overview of Development Options

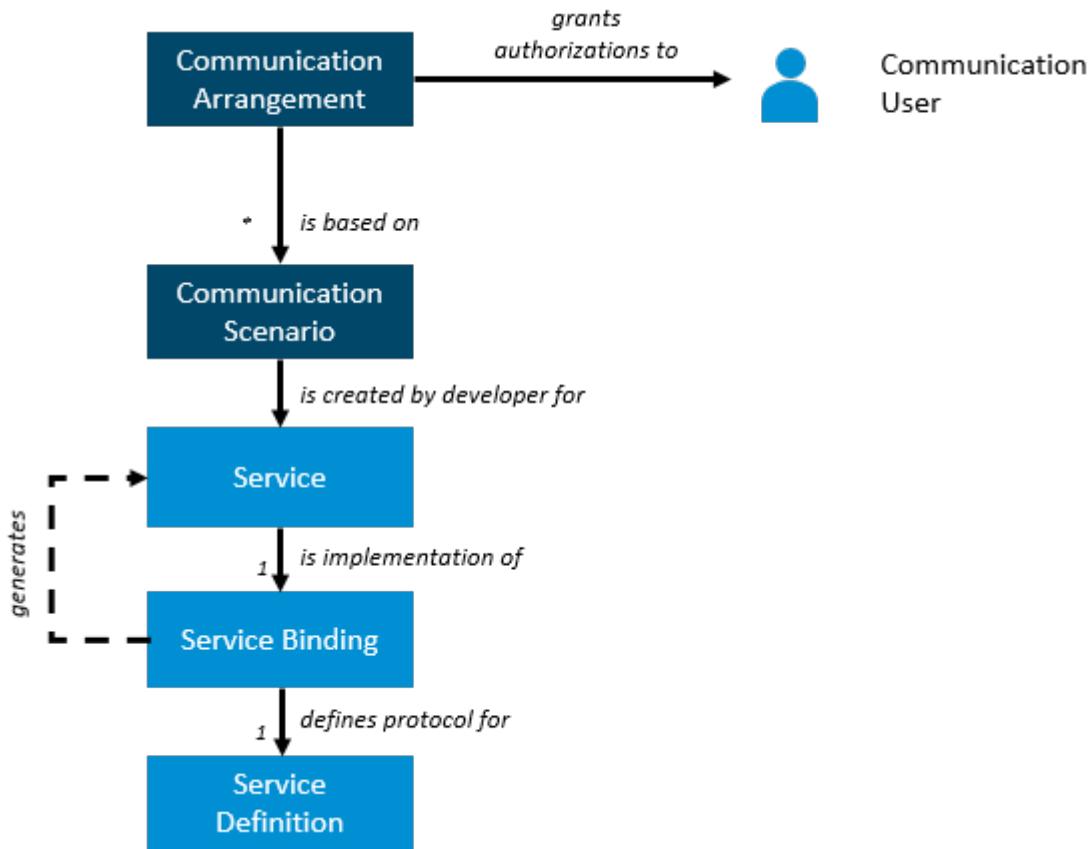
You can provide access the following ways:

- Unrestricted access
- Access based on activities from an authorization object

## 4.2.7.2.5.1 Granting (Unrestricted) Access for Communication Users

In this scenario, you get information about how to grant unrestricted access for communication users with a few steps.

In this scenario, to provide access to the business service, you must create a communication scenario. Administrators then create a communication arrangement based on this communication scenario. Note that all objects that need to be created are highlighted in dark blue in the graphic.



This scenario requires only a few steps to grant authorizations, and it's the minimum that you must do to make your service available to communication users. However, with this scenario, you always grant unrestricted write **and** read access. If you want to fine-tune the authorizations in more detail, check the scenario *Granting Access Based on Activities*.

### 4.2.7.2.5.1.1 Creating a Communication Scenario for Unrestricted Access (Developer)

In ABAP Development Tools, you can define communication scenarios, which can be used to access business services using a technical communication user.

#### Context

In ABAP Development Tools, you define a communication scenario and add the required services to the scenario. An administrator can then use this communication scenario and create a communication arrangement based on it. This communication arrangement is the basis for accessing the business service using a communication user. If you don't include an authorization object to the communication scenario, the communication user will have unrestricted access to the business service.

For more information, see the information about consuming services in the context of APIs in the [user guide for ABAP development tools](#) (ADT).

## Procedure

1. Choose the relevant package in the project explorer.
2. Right-click to open the context menu and choose ► *New* ► *Other* ► *ABAP Repository Object* ▾ to launch the creation wizard.
3. Choose ► *Communication Management* ▶ *Communication Scenario* ▾ and choose *Next*.
4. Enter a name and a description and choose *Next*.
5. Choose *Finish* to create a transport request.
6. On the *Inbound* or the *Outbound* tab, add your inbound or an outbound business service to the communication scenario.
7. If you want to test your communication scenario directly in the development system, publish it locally to make sure that the required services have been generated.

### 4.2.7.2.5.1.2 Creating a Communication Arrangement (Administrator)

To grant authorizations to a communication user, as administrator, you create a communication arrangement. In this example, the communication arrangement is based on the communication scenario created by the developer.

## Prerequisites

You have created a communication user that is used for data exchange with other systems. To create a communication user, you use the *Maintain Communication Users* app on the SAP Fiori launchpad (see also [How to Create Communication Users](#)).

You have created a communication system, which represents the communication partner within a communication. To create a communication system, you use the *Maintain Communication Systems* app (see also [How to Create Communication Systems](#)).

## Procedure

1. Open the SAP Fiori Launchpad.
2. Choose *Communication Arrangements*.
3. Create a new communication arrangement based on the communication scenario, using the communication user and system that you have already created.

For more information, see [How to Create a Communication Arrangement](#).

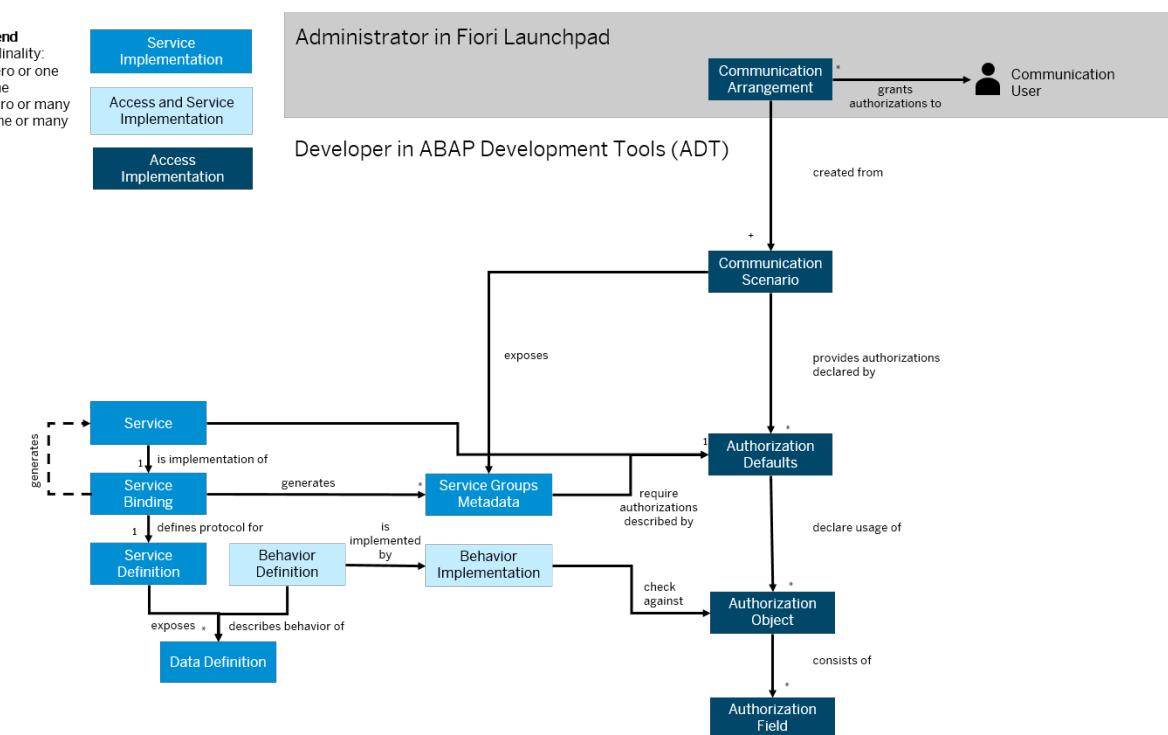
## Results

The communication user in the communication arrangement gets the authorizations that have been defined as part of the communication scenario by the developer. In the example of unrestricted access, no authorization object or fields have been defined, so the communication user gets full access to the business service.

### 4.2.7.2.5.2 Granting Access Based on Activities for Communication Users

In this scenario, you grant access depending on what the communication user should be allowed to do, for example, read or write access.

This scenario is more complex than a simple unrestricted access grant because you also need to create an authorization object and implement authority checks in the behavior implementation.



In this documentation, you get a more detailed overview of how to implement a separate read and write access. You can achieve such a separation by creating an authorization object with the standard authorization field for activities, an authority check to protect the service, and a communication scenario with authorization fields and activities defined. The administrator can then create two communication arrangements for read and write access each, for example.

## 4.2.7.2.5.2.1 Protecting the Service (Developer)

With creating an authorization object for your service and with the implementation of a protection against unauthorized creation and change activities, you implement a basic protection of your service.

After this, you can implement how authorizations are granted to communication users.

### Related Information

[Developing Authorizations for Communication Users \(Developer\) \[page 857\]](#)

## 4.2.7.2.5.2.1.1 Defining Permitted Activities in an Authorization Object

To define the permitted activities for a service, create an authorization object with the standard authorization field *Activity* (ACTVT) for the permitted activities.

### Procedure

To define the permitted activities, follow the procedure for defining authorization objects in the [user guide for ABAP development tools](#) (ADT).

#### ⓘ Note

In our example, in the newly created authorization object, under *Authorization Fields*, you only need the predefined *Activity* authorization field to define the permitted activities for the business users of your service. In our example, these activities would be *01 Add or Create*, *02 Change*, *03 Display*, and *06 Delete* for standard activities. In addition, you need *93 Calculate* for the business object-specific activity *Calculate Bonus*. If you click on an empty row and choose `ENTER`, the list of available values is shown.

In theory, by using the syntax `authorization: update` on an operation in the entity's behavior definition, you could use the authorization check that is implemented for the update operation for the annotated operation. This is also referred to as "authorization delegation". However, for security reasons, we recommend that you carefully consider whether you want to use authorization delegation because in this case, a user who is allowed to update a business object is then also allowed to perform all other activities on the business object.

## Results

In the following, the newly created authorization object for the bonus calculation example is called ZBNSCLC\_AO.

### 4.2.7.2.5.2.1.2 Modeling a Protection Against Unauthorized Write Activities

To protect your service from activities such as create, update or delete by unauthorized users, you can use the authorization controls that are available for services based on managed business objects.

## Context

In the example used here, the service is based on a managed business object *Bonus Calculation*. The use of a managed business object allows you to benefit from the global authorization for managed business objects, which is part of the ABAP RESTful Application Programming model. For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

## Procedure

1. In ABAP development tools, call up the behavior definition of your service.
2. In the behavior definition, define authorization controls for your service.

In the bonus calculation example, they can be as follows:

#### Sample Code

```
managed;
define own authorization context
{
  'ZBNSCLC_AO';
}
define behavior for z_i_bonus_calculation alias calculation
implementation in class zbp_bonus_calculation unique
persistent table zbonusclc
lock master
// Identity and Access Management:
// Enable RAP managed authorization check at create, update, and delete
// -> requires implementation of two methods, one with the addition "FOR
INSTANCE AUTHORIZATION"
// and one with the addition "FOR GLOBAL AUTHORIZATION"
authorization master ( instance, global )
```

With the line `authorization master ( global )`, an authorization check for the standard operations create, update, and delete and for non-standard operations, for example, `calculate bonus`, is

defined. You must implement the authorization check using a method with addition FOR GLOBAL AUTHORIZATION in the behavior implementation.

The sample code also contains the definition of your own authorization context. The own authorization context documents all authorization objects that are used by the business object implementation. In the bonus calculation example used in this documentation, this is the authorization object ZBNNSCLC\_AO because it's checked in the authorization control. The own authorization context is also used to prefill authorization defaults.

### 4.2.7.2.5.2.1.3 Implementing a Protection Against Unauthorized Write Activities

To protect a service such as the example of bonus calculation against unauthorized bonus creations, updates, or deletions, enhance the behavior implementation of the service in ABAP development tools (ADT). The implementation is required by what has been defined in the behavior definition.

#### Context

Among the methods of the behavior implementation, you add methods that check the authorizations for creation and for update and delete.

#### Procedure

1. In the *Source Code Library* folder of your package, under *Classes*, choose the behavior implementation of your service.
2. In the *PRIVATE SECTION* in the handler class of the behavior implementation, define the methods for authorization checks for creation, update, and deletion.

##### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR <behavior definition or  
    its alias> RESULT result.
```

In the bonus calculation example, these method definitions could be the following:

##### ↔ Sample Code

```
METHODS get_global_authorizations FOR GLOBAL AUTHORIZATION  
    IMPORTING REQUEST requested_authorizations FOR calculation RESULT  
    result.
```

3. Add a method implementation that checks the authorizations for creations, updates, and deletions.

## ⓘ Note

For more information about authorization control, see the documentation for the [ABAP RESTful Application Programming Model](#).

In the bonus calculation example, this method could be the following:

### ↔ Sample Code

```
METHOD get_global_authorizations.  
  
    IF requested_authorizations-%create EQ if_abap_behv=>mk-on.  
        * check create authorization  
        AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '01'.  
        result-%create = COND #( WHEN sy-subrc = 0 THEN  
            if_abap_behv=>auth-allowed ELSE  
            if_abap_behv=>auth-unauthorized ).  
    ENDIF.  
    IF requested_authorizations-%update EQ if_abap_behv=>mk-on.  
        * check update authorization  
        AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '02'.  
        result-%update = COND #( WHEN sy-subrc = 0 THEN  
            if_abap_behv=>auth-allowed ELSE  
            if_abap_behv=>auth-unauthorized ).  
    ENDIF.  
    IF requested_authorizations-%delete EQ if_abap_behv=>mk-on.  
        * check delete authorization  
        AUTHORITY-CHECK OBJECT 'ZBNSCLC_AO' ID 'ACTVT' FIELD '06'.  
        result-%delete = COND #( WHEN sy-subrc = 0 THEN  
            if_abap_behv=>auth-allowed ELSE  
            if_abap_behv=>auth-unauthorized ).  
    ENDIF.  
ENDMETHOD.
```

## Results

The implemented methods provide the desired protection from unauthorized uses:

- With the implementation of the method defined with the addition FOR GLOBAL AUTHORIZATION, an unauthorized business user gets an error message when trying to create a business object instance (in the example, a bonus calculation) or when trying to update or delete it via service call.
- Unauthorized business users can't use the *Delete* button in the list view on the service UI. In addition, the *Edit* and *Delete* buttons are hidden in the detail view on the service UI for a business object instance.

Now, the service is secured against unauthorized operations. As a next step, you must make sure that business users who need to use the service get the corresponding authorizations.

## 4.2.7.2.5.2.2 Developing Authorizations for Communication Users (Developer)

After you've protected your service against unauthorized use, you can now create the objects that are needed to grant authorizations for communication users: authorization default values and a communication scenario.

Follow the steps for editing authorization default values and for creating a communication scenario in ABAP Development Tools. You should define own authorization context in the behavior definition first, which can be used to prefill authorization defaults.

### ⓘ Note

For developing authorizations for business users, the procedure is different. For more information, see [Developing Authorizations for Business Users \(Developer\) \[page 824\]](#).

### 4.2.7.2.5.2.2.1 Editing Authorization Default Values

Authorization default values are automatically created when you create a service binding. You can add authorization objects and change the authorization default values.

#### Context

The authorization default values are shown when you include an authorization object into a communication scenario. These values are assigned to communication user if you leave them unchanged. Note, however, you can overwrite the authorization default values by values set in the communication scenario.

For more information about authorization default values, see the [user guide for ABAP development tools](#).

#### Procedure

1. Open the service binding and choose the [Maintain Authorization Default Values](#) link.
2. Choose [Synchronize](#) to add the latest authorization objects from the own context.

### ⓘ Note

The [Synchronize](#) function only **adds** authorization objects that are not yet part of the list of authorization objects; it does not remove objects.

3. To check or change the authorization default values that are assigned automatically, select the authorization objects in the list.  
You can now specify what activities you want to authorize.
4. Choose the authorization object from the list and choose [Default With Field Values](#) from the dropdown list.

In our example, we authorize for the standard activities *Create*, *Display*, *Change*, and *Delete* as well as the business object-specific activity *Calculate*.

## 4.2.7.2.5.2.2.2 Defining a Communication Scenario Including Authorization Values

In ABAP Development Tools, you can define communication scenarios, which can be used to access business services using a technical communication user.

### Context

In ABAP Development Tools, you define a communication scenario and add the required services to the scenario. As part of the communication scenario definition, you also define the relevant authorization fields and activities. An administrator can then use this communication scenario and create a communication arrangement based on it.

#### ⓘ Note

Administrators can't change the authorization activities and values that you define as part of the communication scenario when they create their communication arrangements.

For more information, see the information about consuming services in the context of APIs in the [user guide for ABAP development tools](#) (ADT).

### Procedure

1. Choose the relevant package in the project explorer.
2. Right-click to open the context menu and choose to launch the creation wizard.
3. Choose and choose *Next*.
4. Enter a name and a description and choose *Next*.
5. Choose *Finish* to create a transport request.
6. On the *Inbound* or the *Outbound* tab, add your inbound or an outbound business service to the communication scenario.
7. On the *Authorizations* tab, add the authorization object that you created for the business service.
8. On the *Authorizations* tab, also add the relevant authorization fields and activities. For each authorization field, you can now enter the values that you allow for the communication user. Similarly, you can select which activities are allowed for the communication user.

For example, you could define that only bonus variant V0 can be created using the communication user, but you would allow all possible activities.

The communication user will be defined by the administrator as part of the communication arrangement that is based on this communication scenario.

9. If you want to test your communication scenario directly in the development system, publish it locally to make sure that the required services have been generated.

### 4.2.7.2.5.2.3 Creating a Communication Arrangement (Administrator)

To grant authorizations to a communication user, as administrator, you create a communication arrangement. In this example, the communication arrangement is based on the communication scenario created by the developer, including all authorization restrictions.

#### Prerequisites

You have created a communication user that is used for data exchange with other systems. To create a communication user, you use the *Maintain Communication Users* app on the SAP Fiori launchpad (see also [How to Create Communication Users](#)).

You have created a communication system, which represents the communication partner within a communication. To create a communication system, you use the *Maintain Communication Systems* app (see also [How to Create Communication Systems](#)).

#### Procedure

1. Open the SAP Fiori Launchpad.
2. Choose *Communication Arrangements*.
3. Create a new communication arrangement based on the communication scenario, using the communication user and system that you have already created.

For more information, see [How to Create a Communication Arrangement](#).

#### Results

The communication user in the communication arrangement gets the authorizations that have been defined as part of the communication scenario by the developer. If any authorizations based on authorization fields or activities have been defined or values set in the communication scenario, these also apply.

## 4.2.7.2.6 Providing Read or Write Access to a Business Service Using Privileged Mode

With enablement of privileged mode for a business object, it's possible to consume the business object without the need for additional business authorizations.

### Use Case for Business Users

A typical use case for privileged mode is the execution of an application job. In the bonus calculation example used in this documentation, such an application job could be required to schedule calculations of the target bonus amounts of valid bonus calculations that haven't been calculated yet. For this purpose, you can enable privileged access for the *Bonus Calculation* business object.

As a result, the business user for which the bonus calculation job is scheduled doesn't require all the authorizations that a typical business user of the business service would need. So, there's no need to define an IAM app, a business catalog, and a business role for the authorization to calculate the bonus.

### Use Case for Communication Users

During the execution of instance A of a business object X, write access might be needed to instance B of business object B. This is where privileged mode comes into play. Let's turn to a concrete example: You've created a custom business object for employees and a bonus business object. The employee business object has a performance indicator, such as *Exceeds Expectations*, *Consistently Exceeds Expectations*, and so on.

Now, when the bonus of the employee is calculated using the action *Calculate Bonus*, then the performance indicator of the employee is also updated at the same time, for example, using the action *Update Performance Indicator*. At this point, it makes sense to check whether the user (human or machine) has the authority to calculate the bonus. If so, then the system could also automatically update the performance indicator even if the user doesn't have a dedicated authority for it. What's more, the user might not even be allowed to display the information.

### 4.2.7.2.6.1 Modeling the Authorization Context for Privileged Mode

To enable privileged mode, you need another authorization context named `NoCheckWhenPrivileged`.

#### Context

Let's assume that you want to enable privileged mode for the bonus calculation object that's used as an example in this documentation. First, you define an authorization context called `NoCheckWhenPrivileged`.

in the behavior definition. This authorization context is needed in addition to the own authorization context of the business object that you've already defined in the behavior definition. Then, you enable privileged mode for the business object by disabling `NoCheckWhenPrivileged` using the clause `with privileged mode disabling NoCheckWhenPrivileged`. As a result, authorization objects in the context `NoCheckWhenPrivileged` aren't checked during privileged access.

## Procedure

In the behavior definition, add the following code, for example:

### ↳ Sample Code

```
managed;
define authorization context NoCheckWhenPrivileged
{
    'ZBNNSCLC_AO';
}
with privileged mode disabling NoCheckWhenPrivileged;
define own authorization context
{
    'ZBNNSCLC_AO';
}
define behavior for z_i_bonus_calculation alias calculation
implementation in class zbp_bonus_calculation unique
persistent table zbonusclc
lock master
// Identity and Access Management:
// Enable RAP managed authorization check at create, update, and delete
// -> requires implementation of two methods, one with the addition "FOR
INSTANCE AUTHORIZATION"
// and one with the addition "FOR GLOBAL AUTHORIZATION"
authorization master ( instance, global )
```

## 4.2.7.2.6.2 Providing Read Access Using Privileged Mode

Use the keyword `WITH PRIVILEGED ACCESS` to provide read access using privileged mode.

## Context

For the bonus calculation job in the example used in this documentation, all valid bonus calculations with empty bonus amount are retrieved for calculation in the job. By using the keyword `WITH PRIVILEGED ACCESS` after the entity `z_i_bonus_calculation`, the CDS access controls with checks for authorization objects defined in authorization context `NoCheckWhenPrivileged` are disabled. As a result, unfiltered access to the `z_i_bonus_calculation` entity is possible.

## Procedure

In the example used in this documentation, you can use the following code for selecting bonus calculations:

### ↳ Sample Code

```
SELECT * FROM z_i_bonus_calculation
  WITH PRIVILEGED ACCESS
 WHERE z_i_bonus_calculation~validityenddate >= @sy-datum
   AND z_i_bonus_calculation~bonusamount_v = 0
  INTO TABLE @DATA(lt_bonus_calculation).
```

### 4.2.7.2.6.3 Providing Write Access Using Privileged Mode

Use the keyword `PRIVILEGED` to provide write access using privileged mode.

## Context

After open bonus calculations have been retrieved, the bonus calculation action `calculatebonus` is triggered in the [Bonus Calculation](#) business object with privileged access. By using the keyword `PRIVILEGED` after the behavior definition `z_i_bonus_calculation`, you enable privileged mode. As a result, the bonus calculation is executed without further authorization checks in the RAP authorization control.

## Procedure

In the example used in this documentation, you can use the following code:

### ↳ Sample Code

```
MODIFY ENTITIES OF z_i_bonus_calculation
  PRIVILEGED
    ENTITY calculation
      EXECUTE calculatebonus FROM CORRESPONDING #( lt_bonus_calculation )
      RESULT DATA(result)
      REPORTED DATA(reported)
      FAILED DATA(failed).
```

### 4.2.7.2.7 Troubleshooting

Learn more about apps and tools that help you identify and solve authorization issues.

### 4.2.7.2.7.1 Authorization Trace

There's an app for troubleshooting available: With the *Display Authorization Trace* app you can enable an authorization trace for a business user, which helps you to find out if the user's authorizations are insufficient.

#### More Information

[Display Authorization Trace](#)

### 4.2.7.2.7.2 Debugging

You can use the ABAP Debugger in ABAP development tools to gain more insights about authorization checks. You can do so by activating a breakpoint next to the performed AUTHORITY-CHECK as well as by adjusting the breakpoint activation to be effective for the relevant developer user.

#### More Information

For more information about the ABAP debugger, see the [user guide for ABAP development tools \(ADT\)](#).

## 4.2.8 Business Configuration

Find out how to create custom business configuration objects to enrich your extensibility solutions.

#### Context

An important part of a business application is business configuration. Business configuration in enterprise software refers to a predefined set of configuration options that affect its functionality and behavior. You can use your SAP ABAP Environment to create custom business configuration objects. Find out which tasks a developer (business role SAP\_BR\_DEVELOPER) needs to perform to implement business configuration objects. Mind that lifecycle management is usually performed by a separate user. See [Business Configuration Change Logs \[page 2572\]](#) for more information.

## Procedure

Consider the following to implement your custom business configuration objects:

- Business configuration objects are based on database tables with the delivery type C (@AbapCatalog.deliveryClass : #c). This indicates that the content in these tables can be transported between tenants.
- Business configuration content is usually not directly maintained in a productive system but created in a development system and then transported to the test and productive system. As a result, the corresponding transport handling must also be implemented.
- Business users need a way to maintain content. The number of maintenance objects often exceeds the number of master data or transactional apps, yet at the same time, they are used less frequently. This suggests using a highly standardized way of providing business configuration maintenance apps.
- Business configuration administration often involves the maintenance of language-dependent texts.

To enable the maintenance of content, a full business object is built on top of the table and exposed via an OData service, following the [RAP \(Restful ABAP Programming\)](#) model. The resulting service can be consumed by a frontend application, allowing business users to maintain business configuration content.

SAP offers the [Custom Business Configurations](#) app as a standardized tool for business configuration maintenance. Please refer to [Creating Business Configuration Apps with ABAP RESTful Application Programming Model and Custom Business Configurations App \[page 864\]](#).

As an alternative, you can also enable your business configuration object for upload or download, allowing you to quickly and reliably maintain large amounts of content. In this case, you're not required to develop a full business object. The generic file upload app supports the file format that can be downloaded from SAP ERP or SAP S/4HANA systems, thus supporting an easy migration of data from SAP on-premise systems. For more information, see [here](#).

Should you have special requirements to your maintenance UI, you can also create and deploy your own custom Fiori application. For more information, see [Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#).

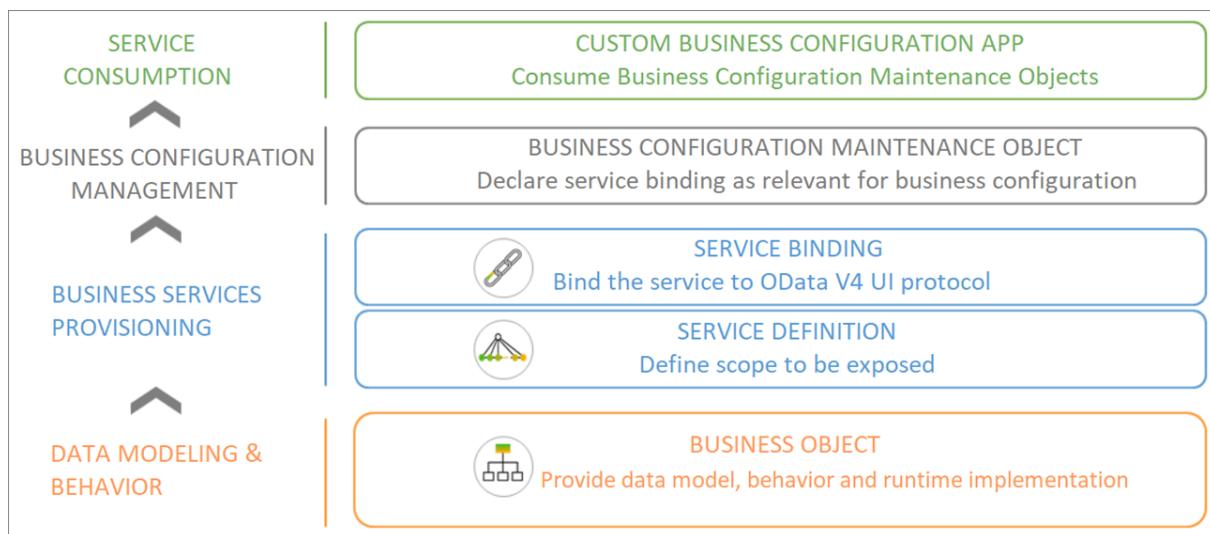
### ⓘ Note

Although the management of customizing transport requests is usually done in the [Export Customizing Transports](#) app by the business process configuration expert, developers can also create such transport requests using the transport organizer view in [ABAP Development Tools](#).

### 4.2.8.1 Creating Business Configuration Apps with ABAP RESTful Application Programming Model and Custom Business Configurations App

As an ABAP Developer you can leverage the [ABAP RESTful Application Programming Model](#) (RAP) and the [Custom Business Configurations](#) (CUBCO) app to create a SAP Fiori UI to maintain customizing table data. You don't need to create your own Fiori app. Everything can be implemented with ABAP repository objects.

For more information, see this [tutorial](#).



The development of a business configuration maintenance Fiori UI requires developers to perform the following fundamental activities:

## 1. Developing Business Object

Based on the customizing tables you define a data model and business object structure. Check the prerequisites of the business configuration maintenance object to see which kind of business object is supported. Using model-driven development approach, you develop actions, validations, and determinations for the entities of the business object to change, maintain and validate configuration entries.

You have the option to use the [Generate ABAP Repository Objects Wizard \[page 869\]](#) to generate all required development objects based on a customizing table. If required, you can then add further functions or enhance the generated business object structure.

For more information, see [Developing a Ready-to-Run Business Object](#) and [Developing Business Logic](#).

## 2. Developing Business Service

Define the scope by exposing the entities of the business object in a service definition and bind the service to OData V4 UI protocol with a service binding. For more information on this, see [Working with Business Services](#).

## 3. Developing Business Configuration Maintenance Object

Declare the service binding as relevant for business configuration by creating a [Business Configuration Maintenance Object \[page 866\]](#).

## 4. Custom Business Configurations App

Based on the Business configuration maintenance object, the custom business configurations app renders a [Fiori Elements List Report Floorplan](#) for the entities exposed by the service binding to maintain the configuration entries. For more information, see [Custom Business Configurations App \[page 2574\]](#).

### 4.2.8.1.1 Business Configuration Maintenance Object

#### Purpose

A business configuration maintenance object declares a service binding as relevant for business configuration. It will be shown on the list of all maintainable business configurations in the Fiori app [Custom Business Configurations App \[page 2574\]](#). The business configuration maintenance object can be maintained via API or ADT. For more information, see also [Creating Business Configuration Apps with ABAP RESTful Application Programming Model](#) and [Custom Business Configurations App \[page 864\]](#).

#### Prerequisites

- The service binding must be of type OData V4 - UI and exposing CDS root entity with draft enabled behavior definition.
- The following two business object composition tree structures are supported:
  - the height of the tree is less or equal than 2.  
Example: Root entity A has two child entities B and C. Entity C has one child entity D. D can't have further child entities
  - the height of the tree is less equal than 3 and the option skip root entity list report is enabled. The root entity must be a singleton and have exactly one child entity.  
Example: Root entity S has one child entity A. Entity A has two child entities B and C. Entity C has one child entity D. D can't have further child entities.
- The data model must consist only of client-dependent tables.

#### Provide Authorizations for a Business Configuration

To grant frontend users the rights to use a business configuration maintenance object, first create an [Identity and Access Management \(IAM\) App](#) and assign it to an [IAM Business Catalog](#). Follow [this user guide](#) but make sure to select the IAM app type [Business Configurations](#) app. In the service tab of the IAM app, select OData V4 as service type and the service binding name of the business configuration maintenance object as service name. In the authorizations tab maintain the authorization object used by the RAP BO for authorization control.

Once you have created the IAM business catalog, assign the catalog to a business role on the Fiori launchpad by following [this user guide](#). Finally, follow [this user guide](#) to assign the business role to business users to grant the rights to use your business configuration inside the *Custom Business Configurations* Fiori app.

## Related Information

[Creating a Business Configuration Maintenance Object in ADT \[page 869\]](#)

[Business Configuration Maintenance Object Cloud Platform API \[page 867\]](#)

[Generating a Business Configuration Maintenance Object with the Generate ABAP Repository Objects Wizard \[page 869\]](#)

### 4.2.8.1.1.1 Business Configuration Maintenance Object Cloud Platform API

Use the ABAP API `mbc_cp_api` to create, update, delete, and read business configuration maintenance objects.

After a business configuration maintenance object has been created, it will be shown on the list of all maintainable business configurations in the Fiori app *Custom Business Configurations* if you have all necessary authorizations regarding the service of the business configuration. You can get the necessary authorizations by following the instructions in [Business Configuration Maintenance Object \[page 866\] > Provide Authorizations for a Business Configuration](#). The end user can then maintain data for the registered business configuration from the frontend.

Before using any of the following methods, first obtain an interface handle by calling `mbc_cp_api->business_configuration_api` with an identifier for your business configuration.

If an error occurs while calling the following methods, an exception of type `cx_mbc_api_exception` will be raised. Use the method `if_xco_news~get_messages` to retrieve all messages that the exception carries.

#### Create a Business Configuration Maintenance Object

Use the `create` method of the obtained business configuration interface handle to create a new object. Supply the following mandatory parameters, as illustrated in the code sample below.

- A name that is shown on the frontend list of all registered business configurations.
- A description for the business configuration that is shown on the frontend.

#### ⓘ Note

The name and description are language dependent. The translations can be maintained using the [Maintain Translations](#) app.

- The OData V4 - UI Service Binding that is used to maintain the business configuration data. The service must be draft enabled.
- The name of the service to be used (as defined in the Service Binding).
- The name of an entity set as exposed by the service definition. This entity set is used as the root node for the UI. Only for this root entity set and its associations a UI is shown. Keep in mind that only one level of sub nodes is supported.

- A transport request of type Workbench to write the business configuration registration to. An entry of type "SMBC" (Business Configuration Object) will be written to that transport request.

#### ↳ Sample Code

```

CLASS zcl_ys_register_bc DEFINITION PUBLIC FINAL CREATE PUBLIC
  INHERITING FROM cl_xco_cp_adt_simple_classrun.
  PROTECTED SECTION.
    METHODS:
      main REDEFINITION.
    ENDCLASS.

CLASS zcl_ys_register_bc IMPLEMENTATION.
  METHOD main.
    DATA(lo_business_configuration) =
      mbc_cp_api->business_configuration_api('/Y123456/HOLIDAY_CALENDAR').
    TRY.
      lo_business_configuration->create(
        iv_name          = 'Holiday Calendar'
        iv_description   = 'A calendar for holidays'
        iv_service_binding = '/Y123456/CAL_I_HOLIDAY_SB'
        iv_service_name  = 'Holiday'
        iv_service_version = 0001
        iv_root_entity_set = 'HolidayRoot'
        iv_transport      = 'X02K900025'
      ).
      CATCH cx_mbc_api_exception INTO DATA(lx_mbc_api_exception).
        DATA(lt_messages) = lx_mbc_api_exception->if_xco_news~get_messages( ).
        LOOP AT lt_messages INTO DATA(lo_message).
          " Use lo_message->get_text( ) to get the error message.
        ENDLOOP.
    ENDTRY.
  ENDMETHOD.
ENDCLASS.

```

The supplied information and an application component are stored in the registration table. The application component is derived from the package that contains the registered service binding. If an error occurs while using the business configuration, it will be shown in an error message on the frontend.

The `create` method also has two optional parameters:

- skip root entity list report: If `abap_true`, the UI automatically navigates to the object page of the root entity skipping the list report. Exactly one root entity must exist.
- app configuration: Configuration of root entity list report and object pages. It is sufficient to maintain only those attributes that should deviate from the standard behavior.

### **Update a Business Configuration Maintenance Object**

You can only update the name and the description of your business configuration by using the respective methods `update_name` and `update_decription`. A transport request of type Workbench must be supplied to transport the updated object.

### **Delete a Business Configuration Maintenance Object**

To delete a business configuration maintenance object, use the method `delete`. A transport request of type Workbench must be supplied to transport the deletion.

### **Check if a Business Configuration Maintenance Object Exists**

To check the existence of a business configuration maintenance object, use the method `exists`. The method has no parameters. It returns `abap_true` if the object exists, `abap_false` if it doesn't.

For an overview of business configuration maintenance objects, use CDS view `I_CustABAPObjDirectoryEntry` with `ABAPObjectType = 'SMBC'`.

### Read a Business Configuration Maintenance Object

To retrieve the details of a business configuration maintenance object, use the method `read`. The method has no parameters. The returned structure contains all the metadata of the business configuration maintenance object.

## 4.2.8.1.1.2 Creating a Business Configuration Maintenance Object in ADT

Find out how to create a business configuration maintenance object using the ABAP Development Tools (ADT).

### Procedure

1. In the *Project Explorer*, select the relevant package node.
2. Open the context menu and choose *New>Other ABAP Repository Object>Business Configuration Management>Business Configuration Maintenance Object>Next* to launch the creation wizard.
3. Enter a name and a description and choose *Next*.

### Related Information

[Working with Business Configuration Maintenance Objects](#)

## 4.2.8.1.1.3 Generating a Business Configuration Maintenance Object with the Generate ABAP Repository Objects Wizard

You can create a business configuration maintenance object together with all related development objects on the basis of a database table by using the *Generate ABAP Repository Objects Wizard*.

### Context

Creating a Fiori app to maintain customizing tables involves many different objects that need to be created manually. On the basis of a database table, this wizard creates all the development objects that are required to maintain the content of this table and, optionally, the content of foreign key associated tables and text tables.

using the [Custom Business Configurations](#) app. If required, you can then, for example, add further functions or enhance the generated business object structure. As an alternative to using the app, you can also build your own custom SAP Fiori elements app based on the generated objects. This is described [in this blog entry](#).

For more information, see [Creating Business Configuration Apps with ABAP RESTful Application Programming Model and Custom Business Configurations App](#). A tutorial on how to use this wizard and the [Custom Business Configurations](#) app is available [here](#).

## Prerequisites

The wizard distinguishes between three types of tables:

- The basis database table in which context the wizard is executed
- (Optional) Additional tables with a foreign key relation of type #KEY to the basis database table
- (Optional) Text tables with a foreign key relationship of type #TEXT\_KEY to one of the tables mentioned above

All tables must fulfill the following requirements:

- have a client key field
- have delivery class C
- allow data maintenance
- (optional) have a timestamp field with data element ABP\_LOCINST\_LASTCHANGE\_TSTMPL. If the table doesn't contain this field, the concurrency control is not active

The basis database table must also meet the following requirements:

- have a timestamp field with data element ABP\_LASTCHANGE\_TSTMPL. For more information, see [RAP Reuse Data Elements](#). If the table doesn't contain this field, the entire ETag is handled by CDS entity I\_CstmBizConfigLastChgd

The additional tables must also meet the following requirements:

- Have a foreign key of type #KEY that fully matches the primary key of the base table

The text tables must also meet the following requirements:

- have a language key field with type LANG
- have a foreign key of type #TEXT\_KEY that fully matches the primary key of one of the other tables

The software component of the target package and the table package must be changeable.

## Procedure

The wizard can't overwrite existing objects or create only a subset of the objects. The generated objects will have the same ABAP language version as the target package.

1. In your ABAP project, open the context menu for a database table and choose [Generate ABAP Repository Objects](#).
2. In the folder [Business Configuration Management](#), select [Maintenance Object](#). Choose [Next](#). Enter the target package and choose [Next](#) again.

3. In the *Configure Generator* page you can define the names and options for the generated objects. The wizard fills all required fields as a proposal based on the description of the basis table. Select *Next*.
4. In the *Preview Generator Output* page, a preview list is available with the objects to be generated. Choose *Next*.
5. Assign a transport request.
6. Choose *Finish* to start the generation of the objects.
7. After the generation process is complete, the generated *Business Configuration Maintenance Object* is shown.

## Procedure Results

The generated RAP business object has the following properties:

- draft enabled, managed
- authorization control based on authorization object S\_TABU\_NAM and the CDS view entity name of the basis table
- concurrency control based on the table timestamp fields
- integration of the BC transport API to validate and record changes on a customizing transport request
- To enable mass editing and manual transport selection, a singleton root entity is part of the generated data model. For more information, see [here](#).
- internal numbering is used for key fields with ABAP type raw(16) (UUID)

## Scenario Options

### Add Copy Action

Select this to include a *Copy* action in the generated app. With this factory action a copy of the selected entry will be created.

### Add Deprecate Actions

Select to include a Deprecate and Invalidate action in the generated app. The table must have a field CONFIGDEPRECATIONCODE with the data element CONFIG\_DEPRECATION\_CODE. With this action a selected entry can be deprecated or invalidated. If a foreign key check is executed on a deprecated or invalidated check table entry, the check returns a warning (deprecated) or error (invalidated). If this option is selected, the user can delete a table entry only if it is in draft mode and the entry was never active.

### Add Data Consistency Check

Select this option to include a *Validation* for the *Prepare* draft action in the generated app, which checks the consistency of field inputs with:

- domains with fixed values
- foreign keys where @AbapCatalog.foreignKey.screenCheck : true

### Copy Parameter Entity

If the option *Add Copy Action* is selected, an abstract entity for the user input is generated. Provide a name for the abstract entity. This is not necessary if all key fields except the client field of the basis table have type ABAP type raw(16).

## Enable Transport Selection Strip

Select to enable a transport selection strip in the header toolbar with the following properties:

- The transport request selection action is only displayed in edit mode.
- The transport request information is displayed in the header toolbar.
- If the save action is executed without a selected transport request and the transport is mandatory, the action to select a transport request is triggered automatically. After selecting a transport request, the save action continues.

If this option is not selected, a transport selection is made possible using a action button. This option is selected by default.

## Transport Selection

Select *Manual* to include the action *Select Transport* in the generated app. With this action, you can select an existing customizing transport request before saving the configuration changes.

Select *Manual with preselection* to include the action *Select Transport* in the generated app. With this action, you can select an existing customizing transport request before saving the configuration changes. When the *Edit* action is performed, a customizing transport request is determined automatically. You can find the determination logic in the ABAP documentation of method `get_transport_request` of interface `if_mbc_cp_rap_tdat_cts`.

Select *No Transport* for an app without a transport option. Suitable for configurations that are to be adjusted in the productive client.

## BC Management

### BC Maintenance Object

Enter the name of the maintenance object.

### Transport Object

You can specify the name of a transport object. If specified, a transport object of type Individual Transaction is generated. Configuration changes are recorded under this transport object instead of as a TABU.

## Data Model

### Tables

This list must contain the basis database table. You can add any additional tables that fulfill the requirements that should be part of the generated RAP business object.

### Text Tables

You can add any text table that fulfills the requirements that are to be part of the generated RAP business object.

## Example

Our example scenario consists of four editable entities with a 1:N cardinality at each level:

- Status Class
- Status Class Text
- Status Code
- Status Code Text

The corresponding database table definitions are as follows:

Status Class:

### ↔ Sample Code

```
@EndUserText.label : 'Status Class'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #C  
@AbapCatalog.dataMaintenance : #ALLOWED  
define table /dmo/statusclass {  
    key client          : abap.clnt not null;  
    key status_class_id : /dmo/status_class_id not null;  
    last_changed_at    : abp_lastchange_tstmp;  
    local_last_changed_at : abp_locinst_lastchange_tstmp;  
}
```

Status Class Text

### ↔ Sample Code

```
@EndUserText.label : 'Status Class Text'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #C  
@AbapCatalog.dataMaintenance : #ALLOWED  
define table /dmo/stclasstext {  
    key client          : abap.clnt not null;  
    key lang            : abap.lang not null;  
    @AbapCatalog.foreignKey.keyType : #TEXT_KEY  
    key status_class_id : /dmo/status_class_id not null  
        with foreign key /dmo/statusclass where client = /dmo/stclasstext.client  
        and status_class_id = /dmo/stclasstext.status_class_id;  
    description         : /dmo/description;  
    local_last_changed_at : abp_locinst_lastchange_tstmp;
```

Status Code:

### ↔ Sample Code

```
@EndUserText.label : 'Status Code'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #C  
@AbapCatalog.dataMaintenance : #ALLOWED  
define table /dmo/statuscode {  
    key client          : abap.clnt not null;  
    @AbapCatalog.foreignKey.keyType : #KEY
```

```

    key status_class_id : /dmo/status_class_id not null with foreign key /dmo/
statusclass where client = /dmo/statuscode.client and status_class_id = /dmo/
statuscode.status_class_id;
    key status_code_id : /dmo/status_code_id not null;
    local_last_changed_at : abp_locinst_lastchange_tstmp;
}

```

Status Code Text:

#### ↔ Sample Code

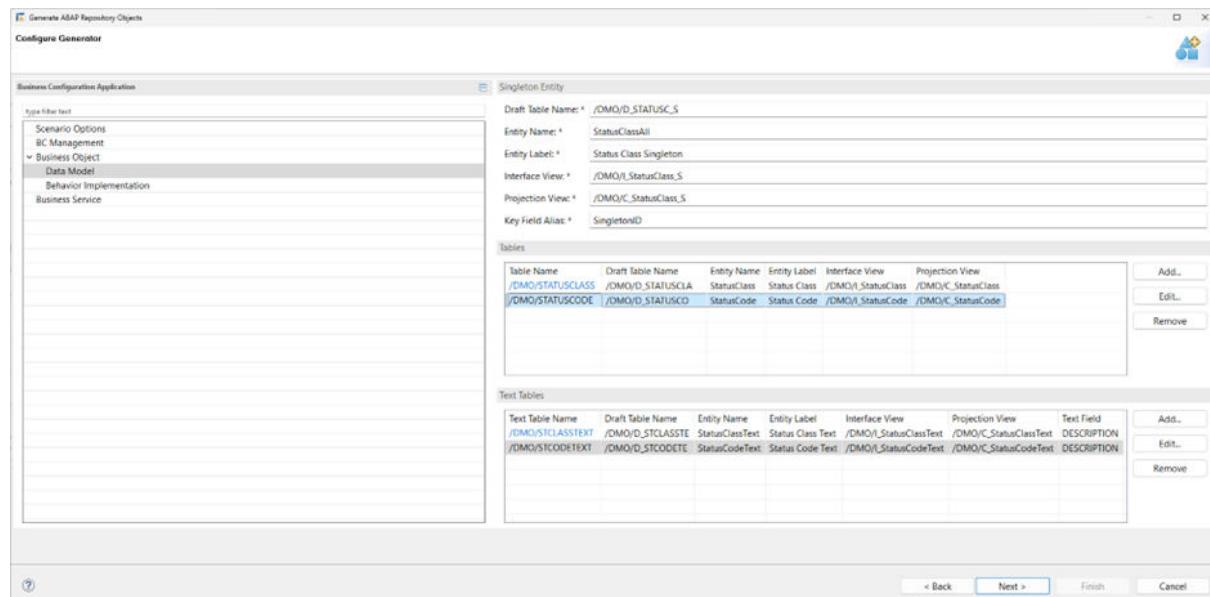
```

@EndUserText.label : 'Status Code Text'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #C
@AbapCatalog.dataMaintenance : #ALLOWED
define table /dmo/stcodetext {
    key client : abap.clnt not null;
    key lang : abap.lang not null;
    key status_class_id : /dmo/status_class_id not null;
    @AbapCatalog.foreignKey.keyType : #TEXT_KEY
    key status_code_id : /dmo/status_code_id not null with
foreign key /dmo/statuscode where client = /dmo/stcodetext.client and
status_class_id = /dmo/stcodetext.status_class_id and status_code_id = /dmo/
stcodetext.status_code_id;
    description : /dmo/description;
    local_last_changed_at : abp_locinst_lastchange_tstmp;
}

```

If you execute the wizard in the context of the table /DMO/STATUSCLASS, the text table /DMO/STCLASSTEXT is automatically added to the list of text tables in the data model section.

Due to the foreign key definition, you can also add the Status Code and Status Code Text table manually.



The resulting table maintenance application then looks like this:

The screenshot shows a Fiori application interface for managing business configurations. At the top, there's a header with the SAP logo and the title "Custom Business Configurations". Below the header, the path "StatusClassAll / 4xx" is displayed, along with a "Delete" button and a refresh icon.

The main area is divided into two tabs: "Status Class" and "Status Code". The "Status Class" tab is active, showing a status class named "4xx" with a status class text of "Status Class: 4xx".

The "Status Code" tab contains a table titled "StatusCodes (2)" with columns "Status Code" and "Description". It lists three entries: "401 Unauthorized", "404 Not found", and a new entry "403 Forbidden" which is currently selected. An "Add Row" button is located at the bottom right of this table.

The "Status Class Text" tab is also present, showing a table titled "StatusClassTexts (1)" with columns "Lang" and "Description". It lists one entry: "English (EN) Client Error". An "Add Row" button is located at the bottom right of this table.

At the bottom right of the entire application area, there is a blue "Apply" button.

## 4.2.8.1.2 Develop Common Capabilities

The guides in this section focus on specific development tasks common to a business configuration application. Whether or not you want to follow these guidelines depends on your initial situation and the desired result of your development.

### Note

The [Business Configuration Maintenance Object wizard \[page 869\]](#) takes these concepts into account. You can therefore use the wizard to generate an executable example which illustrates the implementation of these concepts.

- [Authorization Control \[page 876\]](#)
- [Transport Handling \[page 877\]](#)

## 4.2.8.1.2.1 Authorization Control

### Context

Authorization control in RAP protects your business configuration object against unauthorized access to customizing data.

It's recommended to use the authorization S\_TABU\_NAM together with the CDS root entity name of the business object as table name to perform the read and modify operation. However, you can also check against the physical table names or use a different authorization object instead.

### Procedure

The advantage of using the CDS root entity name is that if the business object is extended by a new table you don't need to adjust the authorization check.

See also [Provide Authorization](#) on how to create the IAM app and add the authorization object.

### Example

The business configuration CDS root entity name is /ITAPC1/I\_Status.

The data control language (DCL) can be defined as follows:

#### Sample Code

```
@MappingRole: true
define role /ITAPC1/I_Status {
    grant select on /ITAPC1/I_Status
    where ( ) = ASPECT pfccg_auth ( 'S_TABU_NAM' , ACTVT = '03' , TABLE = '/ITAPC1/
    I_Status' );
}
```

The global authorization check in the behavior implementation can be defined as follows:

#### ↔ Sample Code

```
METHOD GET_GLOBAL_AUTHORIZATIONS.  
  AUTHORITY-CHECK OBJECT 'S_TABU_NAM' ID 'TABLE' FIELD '/ITAPC1/I_STATUS'  
  ID 'ACTVT' FIELD '02'.  
  DATA(is_authorized) = COND #( WHEN sy-subrc = 0 THEN if_abap_behv=>auth-  
    allowed  
      ELSE if_abap_behv=>auth-unauthorized ).  
  result-%UPDATE      = is_authorized.  
  result-%ACTION-Edit = is_authorized.  
ENDMETHOD.
```

The authorization instance of `S_TABU_NAM` in the IAM app can be defined as follows. Whether or not the user has write or read authorization is finally defined in the General Role Details section of the Business Role the Business Catalog with this IAM app is assigned to.

#### ↔ Sample Code

```
ACTVT = Change, Display  
TABLE = /ITAPC1/I_STATUS
```

## 4.2.8.1.2.2 Transport Handling

### Context

Business configuration content is usually not directly maintained in a productive system, but created in a development system, and then transported to the test system and productive system. As the developer of a business configuration app, you need to ensure that changes to the business configuration content is recorded onto a transport request of type `Customizing` and the user has the option to select the correct transport request.

#### Recording Changes on Transport Request

With the factory class `MBC_CP_API` you can instantiate a transport API to validate and record entity keys of a RAP business object on a transport request.

#### ↔ Sample Code

```
DATA(api) = mbc_cp_api=>rap_table_cts( table_entity_relations = VALUE  
#( ( entity = '/ITAPC1/I_Status' table = '/ITAPC1/STATUS' )  
  ( entity = '/ITAPC1/I_StatusText' table = '/ITAPC1/STATUST' ) ) ).
```

In the behavior implementation define an `Additional Save Implementation` in the header directly after the keyword managed. In the `save_modified` method you can record the changes on the transport request with the method `record_changes` of this API. Check the ABAP Doc of `record_changes` for more information and example implementation.

For each entity of the RAP BO define a [Validation](#) on save for create, update and delete. In the validation implementation call the method `validate_changes` of the API. Check the ABAP Doc of `validate_changes` for more information and example implementation.

### Transport Selection

The before mentioned API methods require a transport request to be passed. For the transport selection define an action with Abstract CDS Entity [`D\_SelectCustomizingTransptReqP`](#) as the parameter. This CDS Entity returns all modifiable transport request of type [`Customizing`](#) where the user either owns the request or owns at least one open task.

Save the result of the action in a virtual field of the draft entity. When recording or validating the changes the previously selected transport request can then be retrieved from the draft entity.

#### Note

When developing custom business configuration objects, you can decide if you want to provide a transport selection screen for change recording. If you are using this transport pattern, we recommend autofilling the transport request based on the software component of the business configuration object.

## 4.2.8.1.3 Develop UI-Specifics

The UI layout of a [\*Business Configuration Maintenance Object\*](#) in the [\*Custom Business Configurations\*](#) app depends on the following layers.

- [CDS Annotations](#)

You can use CDS annotations to define, among other things, the field labels and description, value helps, facets and actions.

- Fiori Elements Settings in the [\*Business Configuration Maintenance Object\* definition](#)

You can configure the list report, object pages and table settings of the [\*Fiori Elements List Report Floor Plan\*](#) rendered by the [\*Custom Business Configurations\*](#) app.

- Generic UI functions of the [\*Custom Business Configurations App\* \[page 2574\]](#).

Grouping of business configuration maintenance objects, intent navigation, display change logs and show documentation are generic UI functions of the [\*Custom Business Configurations\*](#) app.

### 4.2.8.1.3.1 Grouping

The standard variant of the [\*Custom Business Configurations\*](#) app shows you the list of business configuration maintenance objects.

If the list needs to be structured in a simple hierarchical manner, you have the option to group the business configuration maintenance objects semantically.

The screenshot shows the SAP Fiori interface for 'Custom Business Configurations'. At the top, there's a search bar with placeholder 'Name:' and a 'Go' button. Below the search bar is a toolbar with icons for search, help, notifications, and a user profile. The main area displays a table titled 'Custom Business Configurations (4)'. The table has two columns: 'Name' and 'Description'. The data is organized into three sections: 'Configuration Group: ABAP Platform (/ITAPC1/ABAP)', 'Configuration Group: Company Structure (/ITAPC1/ERP)', and 'Configuration Group: Not Assigned (SMBC\_NA)'. Each section contains one item: 'Message Type' (Maintain Message Types), 'Status Code' (Maintain Status Codes), and 'Bonus Type' (Maintain Bonus Types). There are navigation arrows between sections.

As the developer of the business configuration maintenance object, you specify a data element of the same software component as configuration group.

The screenshot shows the 'Business Configuration Maintenance Object: /ITAPC1/STATUSCLASS' dialog. It includes fields for Service Binding, Service Name, Service Version, Root Entity Set, Name, and Configuration Group. The 'Configuration Group' field is set to '/ITAPC1/ABAP'. A checkbox for 'Skip Root Entity List Report' is checked. The 'Configuration Group' field is highlighted with a red border.

The name of the data element can then be used by the custom business configurations app user as the *Group By* element in the table settings.

By clicking on settings you can navigate to the *View Settings* window.

The description of the configuration group data element is shown as the group label. Other attributes of the data element are not relevant. If the configuration group of a business configuration maintenance object is not maintained, the object is assigned to the generic group *Not Assigned*.

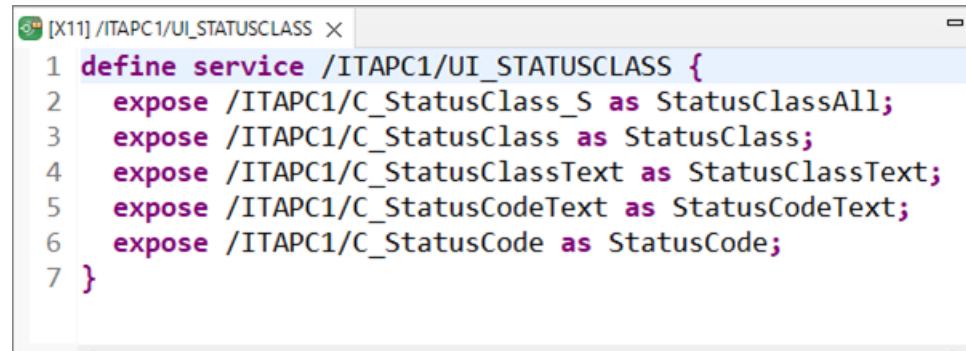
The *Custom Business Configurations* app user can save and share the variant with grouping in the *My Views* dialog.

## 4.2.8.1.3.2 Table Creation Mode

If a new entry is added to a table, the *Custom Business Configurations* navigates to the object page of the new entry by default. This can be changed so that a navigation is not triggered.

This configuration is done with the *Table Creation Mode* attribute in the *Table Settings* section of the [definition](#).

In the following example the business object consists of five entities:



```
[X11] /ITAPC1/UI_STATUSCLASS X
1 define service /ITAPC1/UI_STATUSCLASS {
2   expose /ITAPC1/C_StatusClass_S as StatusClassAll;
3   expose /ITAPC1/C_StatusClass as StatusClass;
4   expose /ITAPC1/C_StatusClassText as StatusClassText;
5   expose /ITAPC1/C_StatusCodeText as StatusCodeText;
6   expose /ITAPC1/C_StatusCode as StatusCode;
7 }
```

For entity *StatusClass* the app shall navigate to the object page when creating a new table entry because the entity has many attributes which the user must maintain. Therefore *Table Creation Mode* is set to *New Page*.

The app shall not navigate to the object page when creating a new table entry for the entity *StatusClassText* because the user only needs to specify language key and text. Therefore *Table Creation Mode* is set to *Creation Row* and *Table Type* to *Grid Table*.

**Business Configuration Maintenance Object: /ITAPC1/STATUSCLASS**

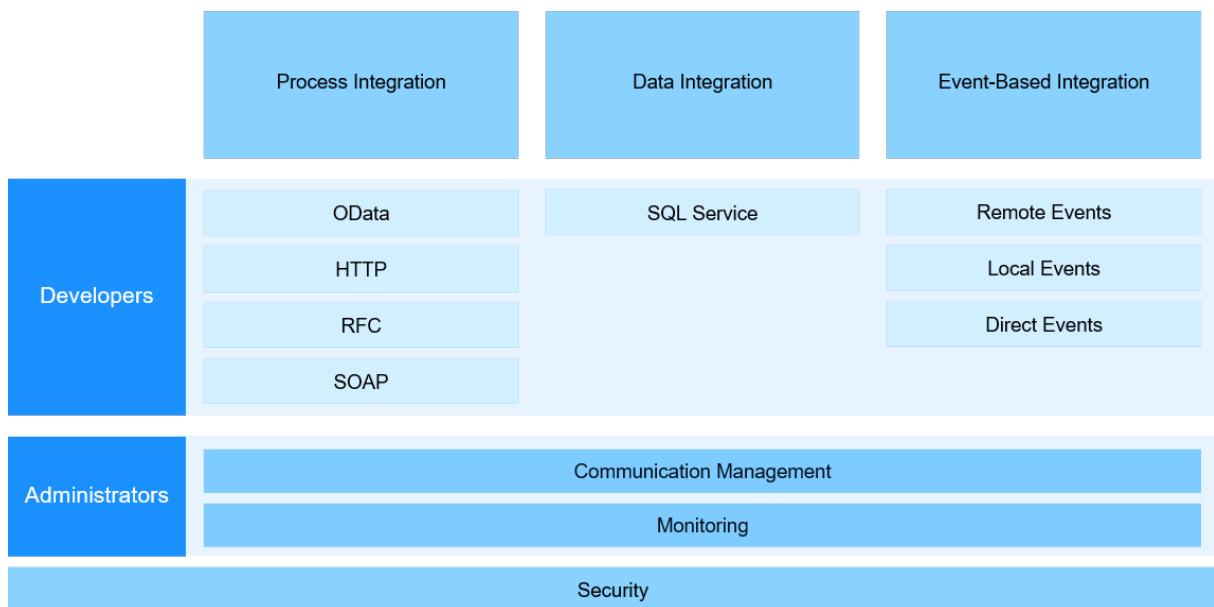
<b>Service Configuration</b>											
Service Binding: *	/ITAPC1/UI_STATUSCLASS_O4	<input type="button" value="Browse..."/>									
Service Name: *	/ITAPC1/UI_STATUSCLASS_O4										
Service Version: *	0001										
Root Entity Set: *	StatusClassAll										
Name: *	Status Code										
Configuration Group:		<input type="button" value="Browse..."/>									
<input checked="" type="checkbox"/> Skip Root Entity List Report											
<b>List Report Configuration</b>											
Initial Load:	Enabled	<input type="button" value="..."/>									
Variant Management:	Page	<input type="button" value="..."/>									
<b>Object Page Configurations</b>											
Entity Set	Section Layout	Variant Management	Editable Header Co...	<input type="button" value="Add..."/>							
StatusClassAll	Page	Control	No	<input type="button" value="Edit..."/> <input type="button" value="Remove"/>							
<b>Table Settings</b>											
Entity Set	Table Type	Selectio...	Table Creatio...	Crea...	Hid...	Hid...	Hid...	Con...	Disa...	Ena...	<input type="button" value="Add..."/>
StatusClass	Responsive Table	Automa...	New Page	No	No	No	No	No	No	No	<input type="button" value="Edit..."/> <input type="button" value="Remove"/>
StatusClassText	Grid Table	Automa...	Creation Row	No	No	No	No	No	No	No	

## 4.2.9 Integration and Connectivity

The ABAP environment enables you to connect to other systems using different protocols. It is possible to consume services from other systems and provide services for consumption in other systems.

ABAP integration and connectivity allows different systems and applications to communicate and work together. In the context of the ABAP environment, this means enabling ABAP applications to exchange data and functionality with other ABAP and non-ABAP systems. Key technologies make this possible, such as APIs, events, web services, and various communication protocols. The goal is to create an integrated IT landscape where applications can seamlessly share information and processes.

These technologies are described in detail with corresponding examples in the [ABAP integration and connectivity](#) development guide.

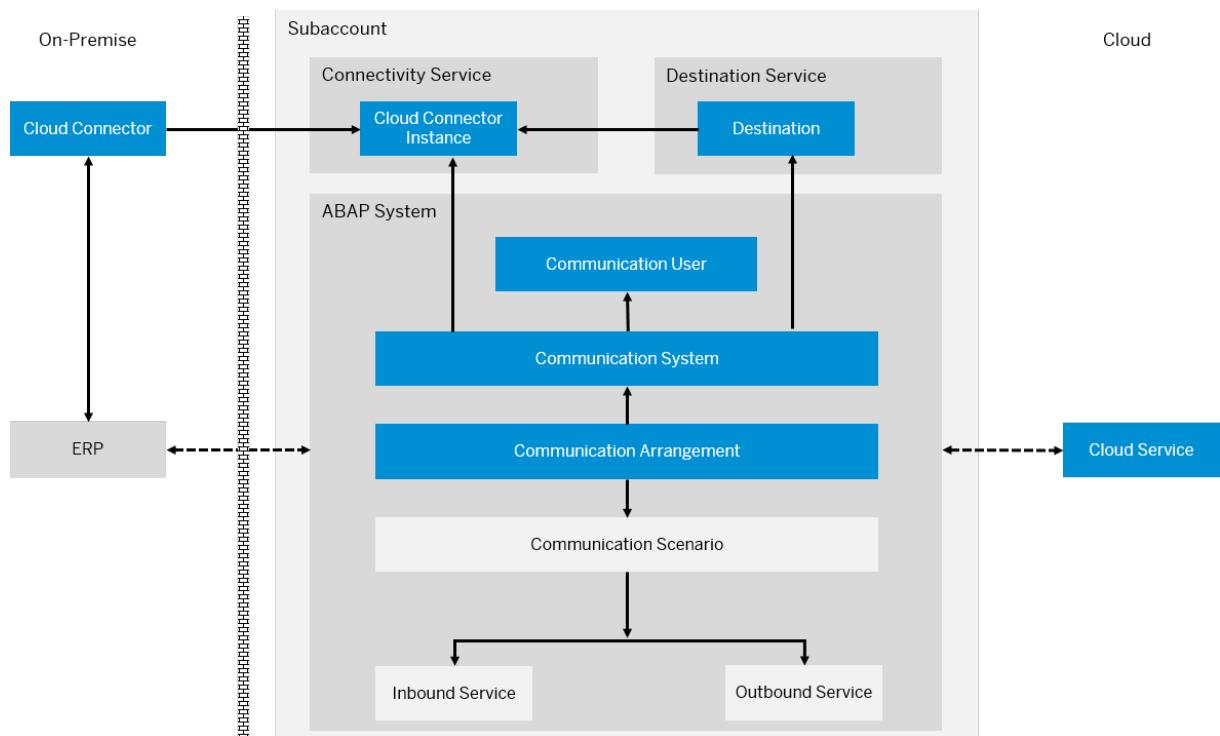


- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/process-integration> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/process-integration]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/data-integration> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/data-integration]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/about-business-events> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/about-business-events]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/odata> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/odata]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/http> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/http]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/rfc> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/rfc]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/soap> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/soap]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/inbound-data-integration-using-sql> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/inbound-data-integration-using-sql]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/remote-events]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/communication-management> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/communication-management]

- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/monitoring> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/monitoring]
- <https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/security-for-abap-integration-and-connectivity> [https://help.sap.com/docs/abap-cloud/abap-integration-connectivity/security-for-abap-integration-and-connectivity]

## 4.2.9.1 Communication Management

Learn more about the basic principles of communication management when integrating your system or solution with other systems to enable data exchange in your ABAP environment.



### 4.2.9.1.1 Communication Scenario

A communication scenario, which is created in the development system using ABAP Development Tools and transported to other systems, is a design time description of how two communication partners communicate with each other. It consists of inbound and/or outbound services as well as supported authentication methods.

It provides technical information, such as the used inbound and outbound services and their service type, for example OData or SOAP, and the number of allowed communication arrangement instances. If the scenario exposes inbound services, it specifies the authorizations that are required to run the services.

The following types of communication scenarios are available:

- **Managed by SAP**, where SAP provides a ready-to-use communication scenario and you create and maintain a communication arrangement.

- **Managed by the customer**, where you develop a communication scenario and create and maintain a communication arrangement.

## Related Information

[Display Communication Scenarios \[page 2595\]](#)

[Overview of Communication Scenarios Managed by SAP](#)

[Developing APIs for Inbound Communication \[page 1034\]](#)

[Outbound Communication](#)

### 4.2.9.1.2 Communication System

A communication system is a specification of a system that represents a communication partner and the technical information required for the communication, such as the host name, port, users for inbound or outbound communication, certificates, etc.

If the communication system represents an on-premise system that is protected by a firewall, the system can be connected by assigning a cloud connector.

Instead of maintaining the credentials for outbound communication in the communication system itself, the communication system can also refer to a destination from the destination service. This can be a destination on subaccount level, or a destination from a destination service instance.

An administrator user in the ABAP environment has to create the communication system in the [Communication Systems \[page 2598\]](#) app in SAP Fiori launchpad. Note that the communication system is not transported between ABAP systems but created locally. The communication partner can vary for each system.

### 4.2.9.1.3 Communication User

A communication user is a specific type of technical user that is assigned to a communication system. The user can be assigned a password or X.509 certificate.

A communication user is added as a user for inbound communication to communication systems.

An administrator user in the ABAP environment creates the communication user in the [Maintain Communication Users \[page 2593\]](#) app in SAP Fiori launchpad. Note that the communication user is not transported between systems but created locally. The technical users and their credentials can vary for each system.

#### 4.2.9.1.4 Communication Arrangement

A communication arrangement is a runtime description of a specific communication scenario. It describes which communication partners communicate with each other in the scenario, and how they communicate.

To describe this runtime behavior, you have to create an arrangement for a scenario, assign the communication system and communication users, and select the authentication method that shall be used.

If the communication scenario exposes inbound services, the communication user is granted the authorizations that have been specified for the communication scenario.

An administrator user in the ABAP environment creates the communication arrangement in the [Communication Arrangements \[page 2595\]](#) app in SAP Fiori launchpad. Note that it is not transported between systems but created locally. For certain SAP-managed scenarios that integrate an SAP BTP service, you can also create a communication arrangement from a service key of the service instance that you would like to connect.

For inbound-only scenarios, you can create a communication arrangement by creating a corresponding service key for the ABAP environment service instance.

#### Related Information

[Maintain a Communication Arrangement for Inbound Communication](#) 

#### 4.2.9.1.5 Destination Service

Using the SAP destination service, you can retrieve and store technical information about the target resource (destination) that you want to connect with your application to a remote service or a system.

The destination service allows you to read and manage the address of a remote service and the user authentication information for the connection on subaccount and service instance level.

#### Related Information

[Destination Service](#)

[Create a Destination \[page 891\]](#)

#### 4.2.9.1.6 Destination

A destination is stored in the SAP destinations service and contains the connection details for the communication partner.

You can use a destination to:

- Connect your application to the Internet (via HTTP, RFC, or WebSocket RFC) or to an on-premise system (via HTTP or RFC)
- Send and retrieve emails by configuring a mail destination

#### ⓘ Note

You can create a destination for a destination service on instance level or subaccount level.

A destination can be referenced in a communication system. To refer a destination from a destination service instance, a communication arrangement for SAP-managed scenario SAP\_COM\_0276 needs to be maintained to enable communication with the destination service instance. See [Create a Destination \[page 891\]](#). In this case, the destination name and the service instance name need to be maintained in the communication system.

## Related Information

[Managing Destinations](#)

### 4.2.9.1.7 Cloud Connector

The Cloud Connector serves as a link between cloud applications and on-premise systems.

In the Cloud Connector, the subaccount of the ABAP system is connected so that the Cloud Connector can be used within a communication system or destination in the subaccount.

It provides an easy setup with a clear configuration of the systems that are exposed to SAP BTP.

## Related Information

[Cloud Connector](#)

[Integrating On-Premise Systems \[page 1184\]](#)

### 4.2.9.2 Supported Protocols and Authentication Methods

Get an overview about supported protocols and authentication methods in the ABAP environment.

Each communication scenario defines which authentication methods can be used. The tables below gives you an overview of all the authentication methods for communication management.

#### ⓘ Note

- The supported protocols and authentication methods only apply to communication scenarios that are managed by customers.

- Since OData is HTTP-based, the listed authentication methods for HTTP are also valid for OData.

## Outbound Communication

Protocol	Available Authentication via Outbound Communication User	Available Authentication via Business Users (Principal Propagation)
HTTP (Internet)	No authentication	OAuth 2.0 SAML Bearer Assertion
	Basic authentication	
	Client certificate authentication	
	OAuth 2.0 Client Credentials grant	
HTTP (Cloud Connector)	No authentication	Not supported. Use the destination service instead. For more information, see <a href="#">Authentication Methods via Destination Service [page 889]</a> .
	Basic authentication	
	Technical user propagation	
SOAP (Internet)	No authentication	Not supported
	Basic authentication	
	Client certificate authentication	
	OAuth 2.0 Client Credentials grant	
SOAP (Cloud Connector)	No authentication	Not supported. Use the destination service instead. For more information, see <a href="#">Authentication Methods via Destination Service [page 889]</a> .
	Basic authentication	
	Technical user propagation	
RFC (Internet)	Basic authentication	Not supported
	Client certificate authentication	
RFC (Cloud Connector)	Basic authentication	Not supported. Use the destination service instead. For more information, see <a href="#">Authentication Methods via Destination Service [page 889]</a> .
SQL service (Internet)	Basic authentication	Not supported

## Inbound Communication

All API requests must be sent to the correct API host. Use the API hostname format:  
**<tenant>.abap.<domain>**.

Protocol	Available Authentication via Inbound Communication User	Available Authentication via Business User (Principal Propagation)
HTTP	Basic authentication	SAML assertion authentication
	Client certificate authentication	OpenID Connect Bearer token
		OAuth 2.0 Authorization Code with Proof Key for Code Exchange (PKCE) OAuth 2.0 SAML Bearer Assertion
RFC (Internet)	Basic Authentication	Not supported
	Client Certificate Authentication	
RFC (Cloud Connector)	Basic Authentication	Not supported
SQL service	Basic Authentication	OpenID Connect Bearer token
	Client Certificate Authentication	

- For more information about the SQL service, see [Accessing ABAP-Managed Data Using SQL Services for Data Integration Scenarios \[page 1198\]](#).
- For more information about SAML Assertion Authentication and OpenID Connect Bearer token, see [How to Create Communication Systems](#).

**① Note**

To use OpenID Connect Bearer token authentication, include the HTTP header **sap-icm-jwt-passthrough** with the value **TRUE** in the request sent to the API host.

- For HTTP protocol: provided that the technical requirements are met, it is recommended to use security sessions. For more information, see [3201227](#).

**① Note**

It is mandatory for the API caller to explicitly request security sessions and confirm that the prerequisites are fulfilled. For more information, see [3319136](#).

- For all logon methods other than *Client certificate authentication*, it is recommended to instruct the server not to use mTLS for logon purposes. For more information, see [3455890](#).

**① Note**

It is recommended that the API caller sends only one type of credential. The evaluation of an unintentionally transmitted client certificate should be deliberately prevented.

## 4.2.9.2.1 Authentication Methods via Destination Service

Get an overview about supported protocols and authentication methods via the destination service in the ABAP environment.

### Outbound Communication

The following table gives you an overview of supported authentication methods for communication management when using the destination service.

Protocol	Available Authentication Methods via Technical User	Available Authentication Methods via Business User (Principal Propagation)
HTTP (Internet)	No authentication	OAuth 2.0 SAML Bearer Assertion
	Basic authentication	OAuth 2.0 User Token Exchange
	Client certificate authentication	
	OAuth 2.0 Client Credentials grant	
HTTP (Cloud Connector)	No authentication	Principal propagation
	Basic authentication	
SOAP (Internet)	No authentication	Not supported
	Basic authentication	
	Client certificate authentication	
	OAuth 2.0 Client Credentials grant	
SOAP (Cloud Connector)	No authentication	Principal propagation
	Basic authentication	
RFC (Internet)	Basic authentication	Not supported
	Client certificate authentication	
RFC (Cloud Connector)	Basic authentication	Principal propagation

#### ⓘ Note

The use of principal propagation isn't supported in the ADT class runner (`if_oo_adt_classrun`) and application jobs. It can only be used during processing of an OData or HTTP service, and only if you execute the service as a business user.

## Related Information

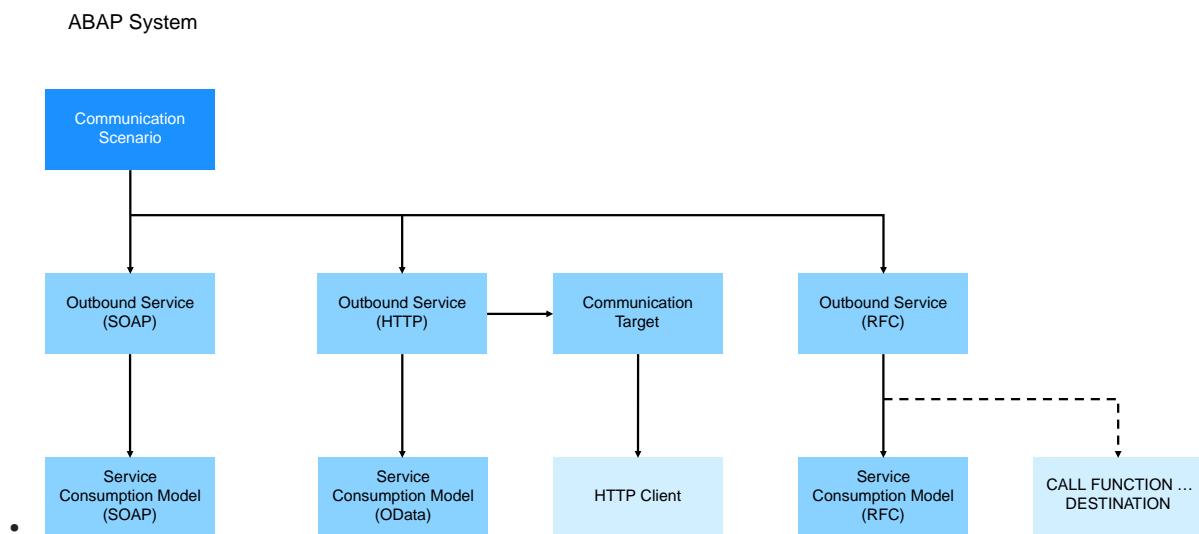
[Supported Protocols and Authentication Methods \[page 886\]](#)

### 4.2.9.3 Developing External Service Consumption (Outbound Communication)

Get more information about consuming external services.

To establish outbound communication between two communication partners, you have to implement the call of a service by doing the following:

- Create an outbound service  
These outbound services can be HTTP (see [Enable HTTP Communication in Your ABAP Code \[page 989\]](#)), RFC (see [Enable RFC Communication in Your ABAP Code \[page 1004\]](#)), or SOAP services (see [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#)).
- For HTTP communication, we recommend to create and use communication targets as described in [HTTP Communication via Communication Targets \[page 990\]](#).
- Create a communication scenario and assign it to the outbound service.  
If you enable multiple arrangements, you can add customer-specific properties to the scenario to enable a simple receiver determination.



## Service Consumption Model

SAP supports various protocols and provides the corresponding clients and functions to handle requests and responses.

HTTP: CL\_WEB\_HTTP\_CLIENT\_MANAGER

RFC: CALL FUNCTION

To simplify the implementation of a remote call, you can create a service consumption model for the external service. The service consumption model creates proxies for the remote service. That way, you can access the service in a strictly typed manner without the need to compile requests and parse responses. The following consumption model types are supported:

- OData
- SOAP
- RFC

## Related Information

[Service Consumption via Communication Arrangements \[page 892\]](#)

### 4.2.9.3.1 Create a Destination

If your business application uses external services, you have to set up a destination for outbound communication either in your subaccount, which is recommended, or in your space.

#### Creating a Destination in Your Subaccount

1. Log on to the SAP BTP cockpit and navigate to your subaccount.
2. In the navigation area, go to and select *New Destination*.
3. Set up your destination and select *Save*. See [Set Up HTTP Communication \[page 1003\]](#), [Set Up RFC Communication \[page 1013\]](#), and [Sending Mails Using SMTP \[page 1032\]](#).

#### Creating a Destination in Your Space

##### ⓘ Prerequisites

- You have created an ABAP service instance. See [Creating an ABAP System](#).
- You have created a destination service instance. See [Creating Service Instances in Cloud Foundry](#).
- You have created a service key. See [Creating Service Keys in Cloud Foundry](#).

1. Log on to the SAP BTP and go to the subaccount that contains the space you'd like to navigate to. See [Navigate in the Cockpit](#).
2. In the navigation area, go to and select the space that contains the ABAP system service instance.
3. Select and navigate to your ABAP system service instance.

4. To open the administration launchpad, click on **...** and select *View Dashboard*.
5. If necessary, provide your logon credentials to access the administration launchpad.
6. In the *Communication Management* section, select the *Communication Arrangements* tile.
7. On the *Communication Arrangements* page, select *New*.
8. In the *New Communication Arrangement* dialog, use the value help to select scenario SAP\_COM\_0276 and give the arrangement a meaningful name (e.g. the name of the destination service instance).
9. Enter the service key of your destination service instance and select *Create*.

**① Note**

The process for connecting to an ABAP service instance URL when creating a new ABAP Cloud project in ABAP development tools for Eclipse has changed. For more information, see [ABAP Service Instance URL \[page 1863\]](#).

### 4.2.9.3.2 Service Consumption via Communication Arrangements

You as a developer can create and expose services for external outbound consumption to a communication partner by creating an outbound service of the required type (HTTP, SOAP or RFC) and specifying the necessary information. To consume these services, you need to create a communication scenario and assign these services to it.

An administrator then creates a communication system and user for the communication partner, maintains a communication arrangement for the scenario using the created communication system, and specifies the authentication method and URLs.

To consume such an external service, you implement the call of the service. Therefore, you need to implement a logical receiver determination in your runtime coding to find the correct target for the outbound call.

The following options are available to establish logical receiver determination provided by communication arrangements:

- *Allowed Scenarios instances: One instance per client*

Only one active communication arrangement can be maintained for the communication scenario in the system. The generated destination can be retrieved by providing the communication scenario ID and the respective outbound service ID.

- *Allowed Scenarios instances: One instance per scenario & communication system*

Multiple active communication arrangements for one communication scenario for different communication systems in one system are enabled. To determine the correct receiver, you must use additional information to find the correct target. You can either do this based on:

- Standard attributes such as *Communication System ID*, or *Business System ID*
- Your own properties, such as plant, country, or company, which can be derived from the business context during runtime. You can define your own properties for the communication scenario and maintain them in the communication arrangements.

The sample codes below show how to find the correct reference to a destination and how to use it when calling a service.

Find the destination based on the communication scenario only:

## ↔ Sample Code

```
data: lr_cscn type if_com_scenario_factory=>ty_query-cscn_id_range.  
* Find CA by Scenario ID  
    lr_cscn = value #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ).  
    data(lo_factory) = cl_com_arrangement_factory=>create_instance( ).  
    lo_factory->query_ca(  
        exporting  
            is_query           = value #( cscn_id_range = lr_cscn )  
        importing  
            et_com_arrangement = data(lt_ca) ).  
    if lt_ca is initial.  
        exit.  
    endif.  
* take the first one  
    read table lt_ca into data(lo_ca) index 1.  
* get destination based on Communication Arrangement and the service ID  
try.  
    data(lo_dest) =  
    cl_http_destination_provider=>create_by_comm_arrangement(  
        comm_scenario   = '<Scenario ID>'  
        service_id      = '<Outbound Service ID>'  
        comm_system_id = lo_ca->get_comm_system_id( ) ).  
  
    catch cx_http_dest_provider_error into  
data(lx_http_dest_provider_error).  
    out->write( lx_http_dest_provider_error->get_text( ) ).  
    exit.  
endtry.
```

Find the destination based on the communication scenario and business system:

## ↔ Sample Code

```
data: lr_cscn  type if_com_scenario_factory=>ty_query-cscn_id_range,  
      lr_bs_id type if_com_system_factory=>ty_query-  
      cs_business_system_id_range.  
* Find CA by Scenario ID & Business System ID  
    lr_cscn = value #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ).  
    lr_bs_id = value #( ( sign = 'I' option = 'EQ' low = '<Business System  
ID>' ) ).  
    data(lo_factory) = cl_com_arrangement_factory=>create_instance( ).  
    lo_factory->query_ca(  
        exporting  
            is_query = value #( cscn_id_range = lr_cscn  
                                cs_business_system_id_range = lr_bs_id )  
        importing  
            et_com_arrangement = data(lt_ca) ).  
    if lt_ca is initial.  
        exit.  
    endif.  
* take the first one  
    read table lt_ca into data(lo_ca) index 1.  
* get destination based on Communication Arrangement and the service ID  
try.  
    data(lo_dest) =  
    cl_http_destination_provider=>create_by_comm_arrangement(  
        comm_scenario   = '<Scenario ID>'  
        service_id      = '<Outbound Service ID>'  
        comm_system_id = lo_ca->get_comm_system_id( ) ).  
    catch cx_http_dest_provider_error into  
data(lx_http_dest_provider_error).  
    out->write( lx_http_dest_provider_error->get_text( ) ).  
    exit.  
endtry.
```

Find the destination based on the communication scenario and a customer-defined property:

#### ↔ Sample Code

```
data: lr_cscn type if_com_scenario_factory=>ty_query-cscn_id_range,
      lt_prpn type if_com_arrangement_factory=>ty_query-ca_property.
* Find CA by Scenario ID & property name/value
  lr_cscn = value #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ).
  append initial line to lt_prpn assigning field-symbol(<fs_prp>).
  <fs_prp>-name = '<Property Name>'.
  append '<Property Value>' to <fs_prp>-values.

  data(lo_factory) = cl_com_arrangement_factory=>create_instance( ).
  lo_factory->query_ca(
    exporting
      is_query           = value #( cscn_id_range = lr_cscn ca_property =
  lt_prpn )
    importing
      et_com_arrangement = data(lt_ca) .
  if lt_ca is initial.
    exit.
  endif.
* take the first one
  read table lt_ca into data(lo_ca) index 1.
* get destination based on Communication Arrangement and the service ID
  try.
    data(lo_dest) =
  cl_http_destination_provider=>create_by_comm_arrangement(
    comm_scenario   = '<Scenario ID>'
    service_id      = '<Outbound Service ID>'
    comm_system_id = lo_ca->get_comm_system_id( ) .
  catch cx_http_dest_provider_error into
  data(lx_http_dest_provider_error).
    out->write( lx_http_dest_provider_error->get_text( ) ) .
    exit.
  endtry.
```

### 4.2.9.3.2.1 Define Specific Properties for Communication Scenarios

For some use cases, such as logical receiver determination based on arbitrary parameters, a communication scenario may require additional parameters that are not part of the standard fields of a communication scenario. Therefore, you can define additional properties in a communication scenario that can be maintained by the administrator in the [Communication Arrangements](#) app. These properties also support secure storage of data, such as passwords or API keys.

#### Prerequisite

A data element that describes your property is required.

The following definitions are derived from the data element:

- *Data Type*
- *Value Help*
- *Field Label* (long text) for the *Communication Arrangements* app

Enter the name of property in the communication scenario and define the following:

- *Default Value*: This value will be added automatically if you create a communication arrangement
- *Data Element*: Describes the property (please see above)
- *Is Multiple*: Indicates that one can maintain multiple values for the property in the *Communication Arrangements* app
- *Is Secure*: Indicates that this property is stored within the secure store. You can use this to store add. Passwords or API Keys
- *Is Hidden*: Indicates that this property is not displayed on the *Communication Arrangements* app and only used for internal usage
- *Value Help*: Indicates that a value help shall be used in the *Communication Arrangements* app

### 4.2.9.3.3 Service Consumption Model

A Service Consumption Model (SRVC) generates proxy artifacts to consume OData, RFC, or SOAP services.

#### Overview

To consume OData, SOAP, or RFC services, you must first create a Service Consumption Model (SRVC).

See [Creating Service Consumption Model](#) for more information. An SRVC creates proxy artifacts and a code snippet that you can copy and paste to your code to perform the service call. For RFC, the SRVC is optional.

### Creating a Service Consumption Model

#### Overview

A remote client proxy request needs a corresponding SRVC.

The benefits of using an SRVC (instead of manually creating a corresponding model) are:

- Simple creation using XML or EDMX file uploading. You don't need to manually define the model.
- The code generator creates code snippets to use as blueprint on how to write client proxy-related coding.

The screenshot shows the SAP SCM interface for a service consumption model named ZBG\_BUPA\_SC. On the left, there's a list of service entity sets, including A\_BupIdentification, A\_AddressHomePageURL, A\_Supplier, etc. On the right, the details for A\_BupIdentification are displayed. It shows the data definition (ZA\_BUPAIDENTIFICATION), behavior definition (ZA\_BUPAIDENTIFICATION), and a code sample for the 'Create' operation. The code sample is written in ABAP and demonstrates how to create a new entity entrydate, country, region, validitystartdate, validityenddate, and authorizationgroup.

```

bpidentificationentrydate = '20170101'
country                   = 'Country'
region                     = 'Region'
validitystartdate          = '20170101'
validityenddate             = '20170101'
authorizationgroup         = 'Authorizationgroup' .

" Navigate to the resource and create a request for the create operation
lo_request = lo_client_proxy->create_resource_for_entity_set( 'A_BUPAIDENTIFICATION'
)->create_request_for_create( ).

" Set the business data for the created entity
lo_request->set_business_data( ls_business_data ).

" Execute the request
lo_response = lo_request->execute( ).

" Get the after image
*lo_response->get_business_data( IMPORTING es_business_data = ls_business_data ).

CATCH /iwbeplcx_cp_remote INTO DATA(lx_remote).
" Handle remote Exception
" It contains details about the problems of your http(s) connection

```

## Example

Create a Service Consumption Model for remote consumption of your OData service. You have the metadata of the corresponding OData service that you want to consume in an XML or EDMX file.

### Step-by-step

See the following information for how to create your SRVC:

- [Generating Proxies for Remote OData Service](#)
- [Generating Proxies for Remote Function Call \(RFC\)](#)
- [Generating Proxies for Remote Web Service](#)

## Constraints

Not all OData service model features are supported for Service Consumption Models. For example, you might not be able to create a Service Consumption Model for your underlying OData service. This is the case if your Service Model contains Complex Collections, Actions, or Functions.

#### 4.2.9.3.4 OData Outbound Communication

The OData Client Proxy is the interface between the client (consumer of a service) and the service implementation (data provisioning) in the OData service consumption in ABAP. This enables ABAP developers to create OData client coding to run OData requests in your ABAP coding.

OData outbound connectivity enables you to consume OData services from other systems. To enable OData outbound communication, there are two different stages that are performed by different roles:

#### Development Tasks

Development tasks are performed by developers. Development tasks are all tasks that are performed within ABAP development tools for Eclipse, such as:

- Creating a Communication Scenario
- Creating an outbound service

For more information, see [Enable OData Communication in Your ABAP Code \[page 898\]](#).

#### Administrator Tasks

Administrator tasks are performed by administrators. Administrator tasks are all tasks that are performed in SAP Fiori launchpad, such as:

- Creating a communication system
- Creating a communication arrangement
- Creating communication users

For more information, see [Set Up OData Communication \[page 901\]](#).

[Enable OData Communication in Your ABAP Code \[page 898\]](#)

[Set Up OData Communication \[page 901\]](#)

To set up OData communication, use the corresponding communication management apps.

[OData Client Proxies \[page 901\]](#)

There are several consumption types and OData versions for your OData Client Proxy configurations, depending on your use case.

[OData Client Proxy Overview \[page 903\]](#)

Get an overview of some of the most common OData Client Proxy tasks.

[OData Client Proxy User Guide \[page 917\]](#)

See examples and steps for using the Data Client Proxy.

[OData Requests \[page 929\]](#)

Learn how to use OData Requests.

## 4.2.9.3.4.1 Enable OData Communication in Your ABAP Code

### Feature Scope

Based on a service metadata file, you can generate the ABAP code for an OData call by using a Service Consumption Model. See [Service Consumption Model as OData Consumer \[page 898\]](#) for more information.

### 4.2.9.3.4.1.1 Service Consumption Model as OData Consumer

A Service Consumption Model is the main requirement for consuming an OData service in the ABAP environment. To learn how to create a Service Consumption Model with ABAP development tools for Eclipse (ADT), see [Creating Service Consumption Model](#).

#### Note

We recommend to create a new package for the Service Consumption Model. Generated objects are then created within this package. This allows better organization and clarity.

A Service Consumption Model generates ABAP code from the service metadata file (EDMX file) of the OData service you want to consume. For every operation (Create, Read, Read List, Update, and Delete), a code snippet is generated that indicates how to perform the corresponding operation.

For more information on creating a Service Consumption Model, see [Generating Proxies for Remote OData Service](#).

### 4.2.9.3.4.1.2 OData Communication via Communication Arrangements

OData outbound communication can be established using so-called communication arrangements.

### Prerequisites

- You've created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).
- You have a service meta data file (EDMX file) for the service you want to consume.
- You have created and activated a service consumption model (SRVC) of type oData. See [Creating Service Consumption Model](#) for more information.

## Procedure

1. Create a corresponding outbound service of type HTTP.
  1. In your ABAP project, select the relevant package node in the Project Explorer. Open the context menu and choose **File > New > Other > ABAP Repository Object > Cloud Communication Management > Outbound Service > Next** to launch the creation wizard.
  2. Select **HTTP Service** in the **Service Type** dropdown list.
  3. Choose Next and select a transport request.
  4. Optional: Go to your outbound service and enter the URL in the field **Default Path Prefix**.
  5. Save the outbound service.
2. Add the newly created outbound service to a communication scenario. See [Service Consumption via Communication Arrangements \[page 892\]](#) for more information.
3. Publish the communication scenario. The administrator can then create the required communication management objects as described in [Next Steps \[page 900\]](#).
4. Call the OData service as described in the example. Copy the code snippet from the **Overview** tab in your SRVC and add the `comm_system_id` and `service_id` parameters if necessary. According to your developed communication scenario, add the following:

<code>comm_scenario</code>	mandatory	ID of the developed communication scenario
<code>comm_system_id</code>	optional	ID of the configured communication system
<code>service_id</code>	optional	ID of the developed outbound service

## Example

The following code is generated by the SRVC for the create operation of the OData service `Business User`.

### ↔ Sample Code

```
DATA:  
  ls_business_data  TYPE business_user=>tys_users,  
  lo_http_client    TYPE REF TO if_web_http_client,  
  lo_client_proxy   TYPE REF TO /iwbep/if_cp_client_proxy,  
  lo_request        TYPE REF TO /iwbep/if_cp_request_create,  
  lo_response       TYPE REF TO /iwbep/if_cp_response_create.  
TRY.  
  * Create http client  
  DATA(lo_destination) =  
    cl_http_destination_provider=>create_by_comm_arrangement(  
    *                                         comm_scenario  = '<Comm  
    Scenario>'  
    *                                         comm_system_id = '<Comm System  
    Id>'  
    *                                         service_id     = '<Service  
    Id>' ).  
  *lo_http_client =  
  cl_web_http_client_manager=>create_by_http_destination( lo_destination ).  
  lo_client_proxy = /iwbep/cl_cp_factory_remote=>create_v4_remote_proxy()
```

```

EXPORTING
    is_proxy_model_key      = VALUE #( repository_id      = 'DEFAULT'
                                    proxy_model_id     = 'BUSINESS_USER'
                                    proxy_model_version = '0001' )
    io_http_client          = lo_http_client
    iv_relative_service_root = '<service_root>' .
ASSERT lo_http_client IS BOUND.
* Prepare business data
ls_business_data = VALUE #(
    id                  = '11112222333344445555666677778888'
    first_name          = 'FirstName'
    last_name           = 'LastName'
    phone               = 'Phone'
    email               = 'Email'
    address             = 'Address' .
" Navigate to the resource and create a request for the create operation
lo_request = lo_client_proxy->create_resource_for_entity_set( 'USERS' )-
>create_request_for_create( ).
" Set the business data for the created entity
lo_request->set_business_data( ls_business_data ) .
" Execute the request
lo_response = lo_request->execute( ) .
" Get the after image
*lo_response->get_business_data( IMPORTING es_business_data =
ls_business_data ) .
CATCH /iwbep/cx_cp_remote INTO DATA(lx_remote).
" Handle remote Exception
" It contains details about the problems of your http(s) connection
CATCH /iwbep/cx_gateway INTO DATA(lx_gateway) .
" Handle Exception
CATCH cx_web_http_client_error INTO DATA(lx_web_http_client_error) .
" Handle Exception
RAISE SHORTDUMP lx_web_http_client_error.
ENDTRY.

```

## Note

We recommend to retrieve the correct destination reference based on the communication scenario and a customer-defined property as described in [Service Consumption via Communication Arrangements \[page 892\]](#).

## Next Steps

To test your outbound call, you have to provide a configuration for the outbound service you want to call in your code.

Create a communication system and communication arrangement for the communication scenario in the Communication Systems and Communication Arrangements apps and maintain the required data.

These tasks are performed by the administrator. For more information, see [Set Up OData Communication \[page 901\]](#).

## 4.2.9.3.4.2 Set Up OData Communication

To set up OData communication, use the corresponding communication management apps.

### Context

The communication management is done by the administrator in SAP Fiori launchpad.

### Procedure

1. Create a communication system. A communication system determines which target system is called and which authentication methods are used. It also provides the user that is required to register at the target system. For more information, see [Communication Systems \[page 2598\]](#).
2. In the communication system, create an outbound communication user.
3. Create a communication arrangement. The communication arrangement is based on the communication scenario, that is created by the developer in ABAP Development Tools for Eclipse. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

### Results

The communication management established a connection to the system from which you want to consume the OData service. You can now perform the service call as described in [OData Communication via Communication Arrangements \[page 898\]](#)

## 4.2.9.3.4.3 OData Client Proxies

There are several consumption types and OData versions for your OData Client Proxy configurations, depending on your use case.

### Consumption Types

The OData Client Proxy can be either local or remote.

#### Local Client Proxy

Use this Client Proxy for the consumption of an OData service on the current server without HTTP.

- No HTTP overhead
- The OData service is processed in the same application session. This allows integration testing, for example, including stubbing.

## Remote Client Proxy

Use this Client Proxy for the consumption of an OData service that is offered on a remote server.

## OData Versions

### OData Version 2 Client Proxy

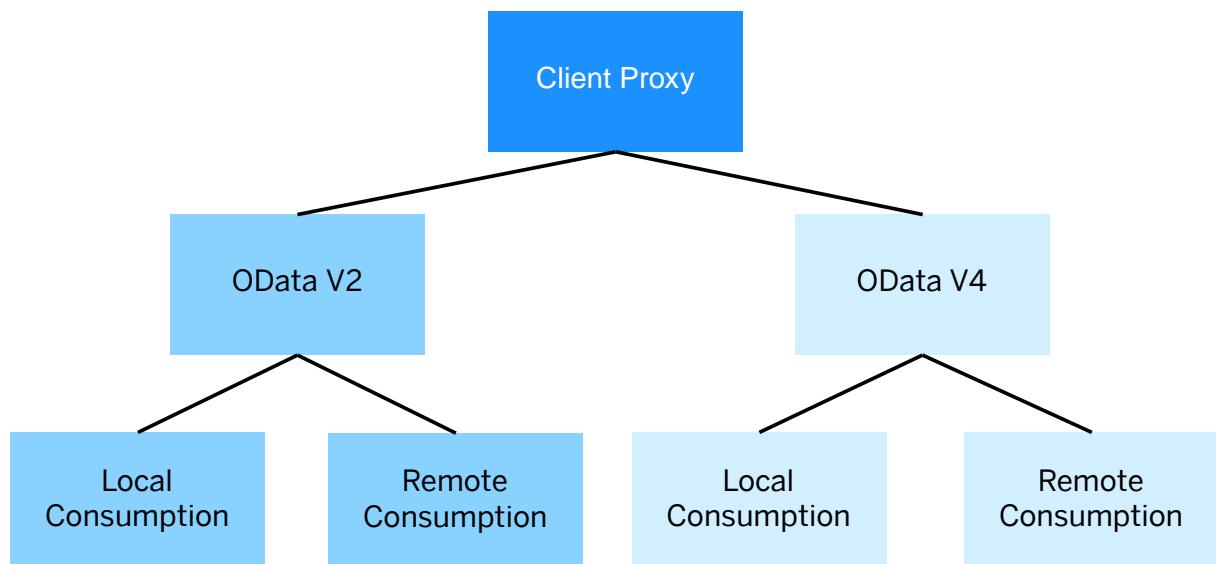
Use this Client Proxy for the consumption of an OData Version 2 service.

### OData Version 4 Client Proxy

Use this Client Proxy for the consumption of OData Version 4 services.

#### *i* Note

These configurations (the OData version and consumption type) depend on each other. For example, there is a local OData Version 2 Client Proxy or a remote OData Version 4 Client Proxy, but not a local Client Proxy independent of the OData version. The asynchronous Client Proxy is only available for remote consumption of OData Version 4 services.



## Constraints

For Client Proxy known constraints, see SAP Note [2428114 - SAP Gateway Foundation SAP\\_GWFND OData Client Proxy - Known Constraints](#).

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Outbound Communication \[page 897\]](#)

### 4.2.9.3.4.4 OData Client Proxy Overview

Get an overview of some of the most common OData Client Proxy tasks.

OData Client Proxy tasks include:

[Client Proxy Instance Types \[page 904\]](#)

Create remote and local Client Proxy instances in OData Version 2 or Version 4.

[Batch and Change Set Request Instance \[page 906\]](#)

Batch requests group multiple individual requests into a single HTTP request payload.

[Deep Create Data Description Node \[page 907\]](#)

In OData Version 4, deep create requests create an entity that includes related entities.

[Expand Node Instance \[page 909\]](#)

An expand node instance causes related entities to be included inline in the response.

[Resource Instance \[page 910\]](#)

A resource instance represents a resource that is shared between applications and identified using URLs and defined in the data model.

[Filter Node Instance \[page 912\]](#)

A filter node instance filters and returns only the results that match the specified expressions.

[Request Instance \[page 913\]](#)

A request instance describes the action you want to take on a resource.

[Response Instance \[page 914\]](#)

A response instance includes information that the request returns.

[Handling Exceptions \[page 916\]](#)

Handling exceptions when running Client Proxy-related coding.

## 4.2.9.3.4.4.1 Client Proxy Instance Types

Create remote and local Client Proxy instances in OData Version 2 or Version 4.

### Client Proxy Versions

#### Client Proxy Version 2 and Version 4

Depending on the OData service you want to consume (Version 2 or Version 4 service), you'll create a Version 2 or Version 4 Client Proxy instance. The underlying OData protocols differ between Version 2 and Version 4. You can't provide a version-agnostic Client Proxy.

### Consumption Types

The decision of using a local or remote Client Proxy instance depends on the current use case.

#### Remote Client Proxy

A remote Client Proxy is used to consume an OData service offered on a remote server. If you want to create a remote Client Proxy instance, the following is required:

- A configured HTTP client instance
- The relative service root pointing to the service document
- Service Consumption model

If you want to create an HTTP client instance, see [Enable HTTP Communication in Your ABAP Code](#).

**Relative Service Root:** The relative service root (combined with the HTTP destination) should point to the service document for the OData service you want to consume. The service document is the top-level representation of an OData service. It lists the entity sets, singletons, and service functions. It is available at `http(s)://host/service/` base URL. You can access the resources in the service document using a URL that is the concatenation of the base URL and the resource URL.

#### Example

You want to consume the OData Version 4 service `TEA_BUSI` in version 2, which is part of the `/IWBEPE/TEA` OData service group and belongs to the service repository `DEFAULT`.

If the absolute URL to fetch the corresponding service document is `http://xyzialb.rot.sap.corp:1234/sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0002/`, the relative service URL is `/sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0002/`.

The remote Client Proxy needs a model object, which is the runtime representation of a model (compared to the model defined by a model provider class (MPC) for an OData service). The Service Consumption Model is needed since we don't have (for example) statically generated OData service specific proxies. Instead, we have

one generic Client Proxy. To handle everything needed to process a client request, the framework needs this information:

- Name and definition of the EDM artifacts
- The mapping from EDM to ABAP names (most importantly for properties)
- ABAP data types
- ABAP conversion exits

A Service Consumption Model defines the characteristics, properties, or parameters (such as, the number or the age of employees).

For detailed examples on how to create a Service Consumption Model, see [Service Consumption Model \[page 895\]](#).

The OData Client Proxy supports SAP Passport to trace and monitor requests between different systems. To send SAP Passport via the header `sap-passport`, you need to take one of the following actions:

- Define the destination type '`H`' (ABAP-ABAP communication) if you instantiated the HTTP client object with `CL_HTTP_CLIENT->CREATE_BY_DESTINATION`.
- Set the attribute `lo_http_client->PROPERTYTYPE_SEND_SAP_PASSPORT = lo_http_client->CO_ENABLED`.

## Local Client Proxy

Use the local Client Proxy when a locally-deployed OData service is consumed. The local Client Proxy delegates a request to the backend component of the corresponding Version 2 or Version 4 runtime without an HTTP overhead. The use case for this scenario is to create Unit Tests, for example, when you want to build an integration test for your service data provider.

### Note

Since the local Client Proxy is designed for writing tests of your data provider, it skips several layers inside the [SAP Gateway Framework](#) and directly calls the data provider class. This has following effects:

- Certain tests that are usually executed during OData request processing are skipped.
- For both Version 2 and Version 4, the business data received from the (local consumption) response is not re-converted for outbound processing.
- When using the Version 4 local client proxy, the business data you enter is not converted for inbound processing. The data provider receives the data exactly how you entered in the request.

For creating a local Client Proxy instance, the service key is required. The service key includes the service repository id (for example, `DEFAULT`), the service id (the ODATA service internal name), and the service id version (for example, `0003`). You don't need an HTTP client instance or a Service Consumption model.

## Creating an Instance

You can create an instance of the Client Proxy using class `cl_web_odata_client_factory`, which provides these static methods:

- `create_v2_local_proxy`
- `create_v2_remote_proxy`

- `create_v4_local_proxy`
- `create_v4_remote_proxy`

#### Note

All returned instances are from type `/iwbepl/if_cp_client_proxy` and offer the same methods. Certain methods are only used by a specific Client Proxy (for example, a remote OData Version 4 proxy) and can raise an exception if the wrong Client Proxy instance is called.

## Functionality

- The Client Proxy instance is the starting point for the OData service consumption with ABAP.
- The Client Proxy instance provides a corresponding [Resource Instance \[page 910\]](#) (for example, an action import or an entity set).
- The Client Proxy instance provides methods for creating `$batch` and delta link requests (with certain delta link-centered utility methods).

## Examples

For examples on how to create and use Client Proxy instances, see [Client Proxy Examples \[page 918\]](#).

## Related Information

[Resource Instance \[page 910\]](#)

[OData Request as Batch Including Changesets \[page 930\]](#)

[OData Request: Delta Link Query Option \[page 965\]](#)

## 4.2.9.3.4.4.2 Batch and Change Set Request Instance

Batch requests group multiple individual requests into a single HTTP request payload.

## OData Version 4 Batch Requests

These individual requests can be included in a batch request:

- Metadata
- Data

- Data Modification
- Action Invocation
- Function Invocation

Batch requests are submitted as a single `HTTP POST` request to the batch endpoint of a service at the `$batch` URL that is related to the service root. Individual requests in a batch request are evaluated according to the same semantics used with an individual request.

In the *Multipart* format, you can group data modification requests or action invocation requests as part of an atomic change set. Operations outside the change set are executed sequentially. Operations in the change set are executed in any order.

#### ① Note

In batch requests, *operation* is an individual request in the batch request payload. A change set request is a part of this type of batch request.

## Creating a Batch Request Instance

A batch request instance is created on the Client Proxy instance. It adds individual requests, executes the batch request, and checks the execution. The batch request instance creates a change set request instance.

After you create a batch request instance, you can define the operations and add them to the batch request or to a change set. The change set request instance is created directly on the batch request instance.

## Related Information

[OData Client Proxy Overview \[page 903\]](#)

[OData Request as Batch Including Changesets \[page 930\]](#)

## 4.2.9.3.4.4.3 Deep Create Data Description Node

In OData Version 4, deep create requests create an entity that includes related entities.

## Overview

A `deep create` is also known as `deep insert`). In this example, you want to create a new customer and the first sales order from this customer. You can first create the new customer and then the first sales order or you can use a `deep create` to create both at the same time:

```
POST http://host/service/Customers
```

This request contains the following payload:

```
{ "CustomerName" : "James Bond",
  "CustomerId" : 007,
  "Customer_2_Salesorders" : {
    "Object" : "Walther PPK",
    "Category" : "Pistols"
  }
}
```

You must use a Data Description node to describe the properties and inline navigations for deep business data for a request. The Data Description node is necessary to describe the structure of your deep business data to the Client Proxy.

A Data Description node is always created on the corresponding create [Request Instance \[page 913\]](#).

### Note

In OData Version 2, complex properties can't have associations, such as navigation properties. You can only define navigation properties in entity types. In OData Version 4, you can define navigation properties for a complex property.

## Creating an Instance

A Data Description node instance is always created in the corresponding [Request Instance \[page 913\]](#).

## Functionality

The Data Description node instances describe the payload for a deep create request to the Client Proxy. This is required to execute any deep create request.

## Example

For examples of how to create and work with Data Description nodes, see [OData Request: Deep Create \[page 961\]](#).

## Related Information

[OData Request: Deep Create \[page 961\]](#)

## 4.2.9.3.4.4.4 Expand Node Instance

An expand node instance causes related entities to be included inline in the response.

### Overview

When reading an entity with a `GET` request, related entities can also be requested using the `$expand` statement on a navigation property. In this example, you want to read the data about customer “Posey” and the open sales orders for this customer (with navigation property “`Customer_2_Salesorders`”). Here is an example of this OData request:

#### ↔ Sample Code

```
GET http://host/service/Customers('Posey')?$expand=Customer_2_Salesorders
```

### Creating an Instance

In the Client Proxy Framework, an expand expression is described by an expand node instance. It is created directly in the corresponding read request instance.

#### ⓘ Note

In OData Version 2, complex properties can't have associations, such as navigation properties. You can only define navigation properties in entity types. In OData Version 4, you can define navigation properties for a complex property.

An expand node instance is always created in the corresponding [Request Instance \[page 913\]](#). These request types can be used to create an expand node:

- read entity
- read entity list
- read entity zero-to-one

### Functionality

The expand node instances add an expand expression into the underlying read request. You can also set the selected properties for each expand node. If not all properties of the expanded entity are returned, compare to `GET http://host/service/Customers('Posey')? $expand=Customer_2_Salesorders($select=Date,Status)`.

## Example

For examples of how to create and work with expand nodes, see [\\$expand Option \[page 972\]](#).

## Related Information

[Request Instance \[page 913\]](#)

### 4.2.9.3.4.4.5 Resource Instance

A resource instance represents a resource that is shared between applications and identified using URLs and defined in the data model.

## Overview

In OData Version 4, a resource is anything in the model that can be addressed (an entity set, entity, property, or operation).

The OData instance creates REST-based data services that allows resources (identified using URLs and defined in an Entity Data Model), to be published and edited by web clients with simple HTTP messages.

For a standard OData request, you must provide a corresponding URL that points to a resource (for example, `entity_list(id='1')` points to a specific resource):

#### Sample Code

```
/sap/opu/odata/sap/srv/entity_list(id='1')/entity
```

#### Note

The resources aren't OData protocol-specific (Version 2 or Version 4).

In the Client Proxy, you don't need to provide the URI. But you need to create an instance of the resource you want to access (for example, a function or a single entity).

## Creating an Instance

Depending on the resource type, the instance creation can vary. Some resources (for example, an entity set or action resources) are created at the [Client Proxy Instance Types \[page 904\]](#). Other resources are created on another resource instance (for example, an entity resource, created on the entity set resource). You can't call an entity resource without providing the corresponding entity set resource.

## Examples

- GET /sap/opu/odata/sap/srv/teams The target resource is an entity set (**teams**). The resource instance is created at the Client Proxy instance. For more information, see [OData Request: Read Entity List \[page 953\]](#).
- GET /sap/opu/odata/sap/srv/order(customer='Jane Doe') The target resource is a single entity (Jane Doe), so the resource instance is created at the parent resource instance (entity set order).

## Resource Types

These resource types are available to users:

- /IWBEPIF\_CP\_RESOURCE\_ACTION: Describes the resource of an **action** that is created on the Client Proxy instance (for action imports) or on an entity (set) resource (for bound actions).
- /IWBEPIF\_CP\_RESOURCE\_ENTITY: Describes the resource of a **single** entity that is created on an entity (set) resource.
- /IWBEPIF\_CP\_RESOURCE\_ENTITY01: Describes the resource of an **optional** entity (the target entity of a zero-to-one navigation) that is created on an entity resource.
- /IWBEPIF\_CP\_RESOURCE\_FUNCTION: Describes the resource of a **function** that is created on the Client Proxy instance (for function imports) or on an entity (set) resource (for bound functions).
- /IWBEPIF\_CP\_RESOURCE\_LIST: Describes the resource of an **entity set** that is created on the Client Proxy instance or an entity resource.

## Functionality

Resource instances are used to create requests (for example, an update request on an entity) or other resource instances (for example, an action resource from an entity list resource). The options that a resource instance offers depends on the different resource types.

## Example

For examples of how to create and work with resources see [Actions and Functions \[page 921\]](#) and [OData Request: Using Navigation \[page 968\]](#).

## Related Information

[OData Request: Read Entity List \[page 953\]](#)

## 4.2.9.3.4.4.6 Filter Node Instance

A filter node instance filters and returns only the results that match the specified expressions.

### Overview

The `$filter` system query option restricts the set of items returned. It allows you to filter a source collection addressed by a request URI. The expression specified with `$filter` is evaluated for each resource in the collection. Only items where the expression is `true` are included in the response.

In this example, you want to read a collection of sales orders and get only the unpaid orders. Here's an example of the OData request:

#### ↔ Sample Code

```
GET http://host/service/SalesOrders?$filter=Payment eq 'open'
```

A filter expression is described by a filter node instance that a filter factory instance creates and can be set into the Client Proxy instance.

#### ⓘ Note

The filter nodes are not OData protocol-specific (Version 2 or Version 4).

### Creating an Instance

A filter node instance is always created by a filter factory instance. You can create a filter factory instance in a read list request instance.

### Functionality

The main functionality of filter node instances is to create a filter expression for a read request on an entity list, which can be integrated into the corresponding request to only show a subset of the complete result.

### Example

For examples of how to create and work with filter nodes, see [\\$filter Option \[page 974\]](#).

## Related Information

[OData Request: Read Entity \[page 946\]](#)

### 4.2.9.3.4.4.7 Request Instance

A request instance describes the action you want to take on a resource.

#### Overview

You can read or modify an OData [Resource Instance \[page 910\]](#) by sending an OData request on this resource. This example reads all entities for the entity set “**teams**”:

##### ↔ Sample Code

```
GET /sap/opu/odata/sap/srv/teams
```

This example deletes the “**order**” entity of customer “**Jane Doe**”:

##### ↔ Sample Code

```
DELETE /sap/opu/odata/sap/srv/order(customer='Jane Doe')
```

A request instance describes the actual operation you want to perform on the corresponding resource. It includes all relevant information (for example, the request body data for updating an entity) for the operation execution.

##### ⓘ Note

The requests are not OData protocol-specific (Version 2 or. Version 4).

## Creating an Instance

Request instances are created on the resource instance the request will act on (for example, a request to update an entity is created on the corresponding entity resource. The only exceptions are \$batch/changeset requests (these can contain requests acting on different resources), and delta link requests, that are both created on the Proxy Instance

### Request Examples

- `/IWBEP/IF_CP_REQUEST_ACTION`: A request to execute an **ACTION** (using action import or as a bound action).

- /IWBEPIF\_CP\_REQUEST\_BATCH: A \$batch request. It can be used to add other requests (for example, a **DELETE** request) and execute them together as part of a \$batch.
- /IWBEPIF\_CP\_REQUEST\_CHANGESET: A request for a **changeset**. This request can be used to add other requests (for example, CREATE request) into one changeset. The changeset request can only be executed as part of a \$batch request (for example, it must be added to a **\$batch** request before executing the request).
- /IWBEPIF\_CP\_REQUEST\_CREATE: A request to **CREATE** an entity.
- /IWBEPIF\_CP\_REQUEST\_DELETE : A request to **DELETE** an entity
- /IWBEPIF\_CP\_REQUEST\_FUNCTION: A request to execute a **FUNCTION** (using function import or as bound function).
- /IWBEPIF\_CP\_REQUEST\_READ: A request to **READ** an entity.
- /IWBEPIF\_CP\_REQUEST\_READ\_01: A request to **READ** an entity WITH a zero-to-one navigation.
- /IWBEPIF\_CP\_REQUEST\_READ\_DLTA: A request to **READ** the delta of a list of entities
- /IWBEPIF\_CP\_REQUEST\_READ\_LIST: A request to **READ** a list of entities.
- /IWBEPIF\_CP\_REQUEST\_UPDATE: A request to **UPDATE** an entity.
- /IWBEPIF\_CP\_REQUEST\_UPDATE\_L : A request to **UPDATE** a list of entities.

## Functionality

Request instances create an OData request and a corresponding [Response Instance \[page 914\]](#). The options a request instance offers depend on the different request types.

## Example

For examples of how to create and work with requests, see the [OData Requests \[page 929\]](#).

### 4.2.9.3.4.4.8 Response Instance

A response instance includes information that the request returns.

## Overview

Most OData requests return a corresponding response body after the request is successful (an exception can be a successful **DELETE** request with HTTP return code 204 with no content).

The Client Proxy response instance contains information returned by the request. You can use it to read all relevant response information (for example, the entity business data).

## ⓘ Note

The responses aren't OData protocol-specific (Version 2 or Version 4).

## Creating an Instance

A response instance is always created by the corresponding request instance when a request is created.

### Response Types

These response types are available:

- /IWBEPEP/IF\_CP\_RESPONSE\_ACTION: **ACTION** request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_CREATE: **CREATE** request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_DELETE: **DELETE** entity request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_FUNCTION: **FUNCTION** request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_READ: **READ** entity request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_READ\_01: **READ** entity (with zero-to-one navigation) request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_READ\_LST: **READ** entity list request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_UPDATE: **UPDATE** entity request response.
- /IWBEPEP/IF\_CP\_RESPONSE\_UPDATE\_L: **UPDATE** entity list request response.

## Functionality

Response instances return the response business data of a successful OData request execution. The options a response instance offers depend on the different response types.

## Example

For examples of how to create and work with responses, see the [OData Requests \[page 929\]](#).

## 4.2.9.3.4.4.9 Handling Exceptions

Handling exceptions when running Client Proxy-related coding.

### Overview

There are two types of exceptions in the Client Proxy context:

- `/IWBEPCX_GATEWAY`: for client issues (for example, if the provided entity set name is unknown or if you try to fetch business data from a response instance that doesn't have any response data).
- `/IWBEPCX_CP_REMOTE`: for server issues in remote scenarios (for example, the returned HTTP status code is not **2xx** (internal server errors [500] or missing authorizations [401]).

The remote server exception contains this information:

- HTTP Status Code
- HTTP Status Reason Message
- OData Error Details
- HTTP Client Error Details
- HTTP error body
- Content type
- HTTP method

## Handling Exceptions

### Overview

For the **remote** client proxy: always catch both exceptions (`/IWBEPCX_GATEWAY` and `/IWBEPCX_CP_REMOTE`) separately and implement corresponding exception handling.

For the **local** client proxy: catch `/IWBEPCX_GATEWAY`. The remote exception isn't raised in the local scenario.

### Example

Use remote client proxy to consume an OData request. Include proper exception handling in your coding:

```
DATA: lv_http_status_code      TYPE /iwbepcx_gateway=>ty_http_status_code,
      lv_http_status_message  TYPE string,
      lv_error_body           TYPE string,
      ls_odata_error          TYPE /iwbepcx_cp_remote=>ty_s_odata_error,
      lv_client_error_text    TYPE string,
      lx_remote               TYPE REF TO /iwbepcx_cp_remote,
      lx_gateway              TYPE REF TO /iwbepcx_gateway.

TRY.
  "Your Client Proxy Coding
  catch /iwbepcx_cp_remote into lx_remote.
  lv_http_status_code = lx_remote->http_status_code.
  lv_http_status_message = lx_remote->http_status_message.
  lv_error_body = lx_remote->http_error_body.
  ls_odata_error = lx_remote->s_odata_error.
  "Further error handling
  catch /iwbepcx_gateway into lx_gateway.
```

```
lv_client_error_text = lx_gateway->get_text( ).  
"Further error handling  
..."
```

## Steps

**Step 1:** Catch the remote exception and fetch the information on the exception (in addition to your own exception handling):

```
DATA: lv_http_status_code      TYPE /iwbep/cx_gateway=>ty_http_status_code,  
      lv_http_status_message  TYPE string,  
      lv_error_body          TYPE string,  
      ls_odata_error         TYPE /iwbep/cx_cp_remote=>ty_s_odata_error,  
      lx_remote              TYPE REF TO /iwbep/cx_cp_remote.  
CATCH /iwbep/cx_cp_remote INTO lx_remote.  
lv_http_status_code = lx_remote->http_status_code.  
lv_http_status_message = lx_remote->http_status_message.  
lv_error_body = lx_remote->http_error_body.  
ls_odata_error = lx_remote->s_odata_error.  
"Further error handling  
..."
```

**Step 2:** Catch the gateway exception and fetch the corresponding error text (in addition to your own exception handling):

```
DATA: lx_gateway           TYPE REF TO /iwbep/cx_gateway,  
      lv_client_error_text TYPE string.  
CATCH /iwbep/cx_gateway INTO lx_gateway.  
lv_client_error_text = lx_gateway->get_text( ).  
"Further error handling  
..."
```

## 4.2.9.3.4.5 OData Client Proxy User Guide

See examples and steps for using the Data Client Proxy.

The OData Client Proxy User Guide includes:

### [Client Proxy Examples \[page 918\]](#)

OData Proxy examples for OData Version 2 and Version 4 for local and remote client proxies.

### [Actions and Functions \[page 921\]](#)

Create an OData request to run an operation (an action or function) in the Client Proxy instance.

## Related Information

### [OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.5.1 Client Proxy Examples

OData Proxy examples for OData Version 2 and Version 4 for local and remote client proxies.

### Overview

For information about the Client Proxy Instance, see [Client Proxy Instance Types \[page 904\]](#).

## OData Version 2: Local Client Proxy

### Overview

You can create a Client Proxy instance with class `IWBEP/CL_CP_CLIENT_PROXY_FACTORY`. It offers the static method `CREATE_V2_LOCAL_PROXY` to create a local V2 Client Proxy instance. The service key parameters (`service_id` and `service_version`) are required for the V2 OData service you want to consume. You can also specify if you want to write a workload trace (you can later check in transaction `STAD`).

#### ⓘ Note

A workload trace is not written by default.

### Example

Create a Client Proxy instance to consume the OData V2 service “`ODATA_V2_TEST_SERVICE`” in version number 1:

#### ↗ Sample Code

```
DATA: lo_client_proxy TYPE REF TO /iwbp/if_cp_client_proxy.  
      lo_client_proxy =  
      cl_web_odata_client_factory->create_v2_local_proxy( VALUE #( service_id =  
      'ODATA_V2_TEST_SERVICE' service_version = '0001' ) ).
```

## OData Version 2: Remote Client Proxy

### Overview

You can create a Client Proxy instance using class `CL_WEB_ODATA_CLIENT_FACTORY`. It offers the static method `CREATE_V2_REMOTE_PROXY` to create a remote V2 Client Proxy instance. The relative service root parameter (`service_definition_name`) and a configured HTTP web client instance are required.

For more information about how to create a configured HTTP web client instance, see:

- [Enable HTTP Communication in your ABAP Code \[page 989\]](#).
- [HTTP Communication via Communication Arrangements \[page 994\]](#).

The service definition is automatically generated when you create a Service Consumption Model.

The concatenated HTTP destination and the relative service root must point to the service document of the OData service. Depending on your HTTP web client instance configuration, the way your relative service root

looks can vary. In our example, we assume that your HTTP destination doesn't have a path prefix (for example, doesn't point to a specific OData service). The relative service root follows the general pattern /sap/opu/odata/<service\_id>.

## Example

Create a Client Proxy instance to consume the OData Version 2 service with the service definition ODATA\_V2\_SERVICE\_DEFINITION and the service id ODATA\_V2\_SERVICE.

### ↳ Sample Code

```
DATA: lo_client_proxy TYPE REF TO /iwbepl/if_cp_client_proxy,  
      lo_web_http_client TYPE REF TO if_web_http_client.  
    lo_client_proxy = cl_web_odata_client_factory->create_v2_remote_proxy(  
      iv_service_definition_name = 'ODATA_V2_SERVICE_DEFINITION'  
      io_http_client = lo_web_http_client  
      iv_do_fetch_csrf_token = abap_true  
  
      iv_relative_service_root = '/sap/opu/odata/ODATA_V2_SERVICE' ).
```

### ⓘ Note

In this example, the Client Proxy automatically fetches a cross-site request forgery (CSRF) token. It does a HEAD request to fetch a CSRF token. This token is automatically reused for all requests executed with this Client Proxy instance. If you want to change this behavior, set iv\_do\_fetch\_csrf\_token to abap\_false.

## OData Version 4: Local Client Proxy

### Overview

Create a Client Proxy instance using class CL\_WEB\_ODATA\_CLIENT\_FACTORY. It offers the static method CREATE\_V4\_LOCAL\_PROXY to create a local Version 4 Client Proxy instance. The service key parameters (**service\_id**, **relative\_service\_root**, which is “**SRVD**”, and the **service\_version**) are required for the Version 4 OData service you want to consume.

### ⓘ Note

The local Version 4 Client Proxy can't be used to consume SAP OData services. You can only use it to consume your own services.

## Example

Create a Client Proxy instance to consume the OData Version 4 service ODATA\_V4\_TEST\_SERVICE in version number 1:

### ↳ Sample Code

```
DATA: lo_client_proxy TYPE REF TO /iwbepl/if_cp_client_proxy.  
      lo_client_proxy = /  
    cl_web_odata_client_factory->create_v4_local_proxy( VALUE #( service_id  
      = 'ODATA_V4_TEST_SERVICE'
```

```
    service_version = '0001'  
    repository_id = 'DEFAULT' ) ).
```

## OData Version 4: Remote Client Proxy

### Overview

You can create a Client Proxy instance using class `CL_WEB_ODATA_CLIENT_FACTORY`. It offers the static method `CREATE_V4_REMOTE_PROXY` to create a remote Version 4 Client Proxy instance. The relative service root parameters (`relative_service_root`) definition and a configured HTTP web client instance are required.

The concatenated HTTP destination and the relative service root must point to the OData service document. Depending on your HTTP web client instance configuration, the way your the relative service root looks can vary. In our example, we assume that your HTTP destination does not contain a path prefix (i.e. does not point to a specific OData service); then, the relative service root follows the general pattern `/sap/opu/odata4/<service group id>/<repository id>/<service id>/<service version>`

### Example

You want to create a Client Proxy instance to consume the OData Version 4 service with the service definition named `ODATA_V4_SERVICE_DEFINITION` and the service id `ODATA_V4_SERVICE` (version 1), stored in service group `ODATA_GROUP` and repository `SRVD`.

#### Sample Code

```
DATA: lo_client_proxy TYPE REF TO /iwbepl/if_cp_client_proxy,  
      lo_web_http_client TYPE REF TO if_web_http_client.  
      lo_client_proxy =  
        cl_web_odata_client_factory->create_v4_remote_proxy(iv_service_definition_name  
        = 'ODATA_V4_SERVICE_DEFINITION'  
                                         io_http_client  
        = lo_web_http_client  
        iv_do_fetch_csrf_token = abap_true  
        iv_relative_service_root = '/sap/opu/odata4/ODATA_GROUP/SRVD/ODATA_V4_SERVICE/  
        0001' ).
```

#### Note

In this example, the Client Proxy automatically fetches a cross-site request forgery (CSRF) token. It does a `HEAD` request to fetch a CSRF token. This token is automatically reused for all requests executed with this Client Proxy instance. If you want to change this behavior, set `iv_do_fetch_csrf_token` to `abap_false`.

## Constraints

The local Client Proxy (both Version 2 and Version 4) can't be used to consume SAP OData services.

## Related Information

[Service Consumption Model \[page 895\]](#)

### 4.2.9.3.4.5.2 Actions and Functions

Create an OData request to run an operation (an action or function) in the Client Proxy instance.

#### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

##### Function

A Service Operation is represented by a Function Import and can return:

- Primitive type
- A single Entity-Type instance
- A single Complex-Type instance
- Collection of Complex-Type instances
- Collection of Primitive types

The corresponding Function Import can have side effects and can be invoked by any pre-defined HTTP method.

#### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

##### Function

**Functions** are operations exposed by an OData service that don't have side effects. Functions must return data and can include additional path segments. Functions are invoked using HTTP method `GET`.

##### Action

**Actions** are operations exposed by an OData service that can have side effects. Actions can return data but **must not** be composed with additional path segments. Actions are invoked using HTTP method `POST`.

## Operation

Functions and Actions are operations that can return data. Operations are either **bound** to a resource (for example, an entity type), that makes them members of that instance type. Operations can also be **unbound**. Unbound operations are called as static operations (using “action imports” or “function imports”) since a static (unbound) operation can’t be called directly.

## Example Requests

### Version 4 Function Import

“`GetEmployeeByManagerId`” with non-binding parameter “`ManagerId`”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/
GetEmployeeByManagerID(ManagerID='0001')
```

### V4 Bound Action

“`AcChangeTeamOfEmployee`” with non-binding parameter “`TeamId`”

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES('2')/
SAP__self.AcChangeTeamOfEmployee
```

Request body JSON:

```
{
  "TeamID" : "TEAM_02"
}
```

### Version 2 Function

“`SetEmployeeSalary`” with non-binding parameter “`Id`” and “`Amount`”

```
PUT /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/SetEmployeeSalary?
Id='0001'&Amount=200
```

For examples and steps for Import and Bound operations, see:

- [Import Operation \[page 923\]](#)
- [Bound Operation \[page 926\]](#)

## Constraints

- If-Match header is only supported for **remote** consumption.
- For function imports, the If-Match header is only supported for OData **Version 2** requests.
- Composable Functions are **not** supported.
- `$expand` with operations is **not** supported.
- `$select` is **not** supported for actions.
- `Return-Prefer` header is **not** supported for actions.

- System query options are **not** supported for Version 4 functions.
- If the return type of a Version 2 function is an entity type and this entity type is used by more than one entity set, the Version 2 request context only contains the (EDM) name of the first entity set with the underlying entity type. A precise mapping is currently not possible. For remote consumption, this can be handled in the proxy model by setting the correct entity set via method `. /iwbepl_if_v4_med_func_imp->set_entity_set_name`

## 4.2.9.3.4.5.2.1 Import Operation

### Overview

The starting point for an operation import is the corresponding operation resource that is created directly on the Client Proxy instance. You can use the resource instance to create a request instance. You can use the response instance to retrieve the response business data of the successfully executed request. This is fetched from the request instance.

#### ① Note

In OData Version 2, all Operation requests must be created as Function Import requests.

### Example

To invoke the Version 2 function '`PromoteEmployee`' with the non-binding (for example, input) parameters "`Id='0001'`" and "`Level=2`":

```
POST /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/PromoteEmployee?Id='0001'&Level=2

TYPES:
BEGIN OF tys_parameter,
id TYPE /iwbepl_if_cp_client_proxy_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name,
level TYPE i,
END OF tys_parameter.
DATA: lo_client_proxy      TYPE REF TO /iwbepl_if_cp_client_proxy_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name,
      lo_function_resource TYPE REF TO /iwbepl_if_cp_resource_function_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name,
      lo_function_request  TYPE REF TO /iwbepl_if_cp_request_function_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name,
      lo_function_response  TYPE REF TO /iwbepl_if_cp_response_function_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name,
      ls_parameter          TYPE tys_parameter,
      ls_busi_data          TYPE /iwbepl_if_cp_response_function_if_v4_med_func_imp-if_v4_med_func_imp->get_entity_set_name.
      lo_function_resource = lo_client_proxy->create_resource_for_function( 'PROMOTE_EMPLOYEE').
      ls_parameter = VALUE #( id = '0001' level = 2 ).
      lo_function_resource->set_parameter( ls_parameter ).
      lo_function_request = lo_function_resource->create_request( ).
      lo_function_request->set_if_match( '2407' ).
      lo_function_response = lo_function_request->execute(/iwbepl_if_cp_request_function_if_v4_med_func_imp->gcs_http_method-post).
      CHECK lo_function_response->has_business_data( ) = abap_true.
      lo_function_response->get_business_data( IMPORTING ea_response_data = ls_busi_data ).
```

## Steps

**Step 1:** Create the function resource “**PROMOTE\_EMPLOYEE**” with the **internal** name of function “**PromoteEmployee**”:

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_function_resource TYPE REF TO /iwbep/if_cp_resource_function.
      lo_function_resource = lo_client_proxy-
>create_resource_for_function( 'PROMOTE_EMPLOYEE' ).
```

**Step 2:** Set the non-binding (for example, input) parameters on the function resource instance. You can do this on the function resource, using method “**SET\_PARAMETER**”:

```
TYPES:
BEGIN OF tys_parameter,
id TYPE /iwbep/tea_employee_id,
level TYPE i,
END OF tys_parameter.
DATA: ls_parameter TYPE tys_parameter.
ls_parameter = VALUE #(id = '0001' level = 2 ).
lo_function_resource->set_parameter( ls_parameter ).
```

**Step 3:** Create a function request in the function resource instance:

```
DATA: lo_function_request TYPE REF TO /iwbep/if_cp_request_function.
lo_function_request = lo_function_resource->create_request( ).
```

**Step 4:** Select the corresponding HTTP method to invoke the function on the request object. In this example, it is **POST**. You must do this in method **SET\_HTTP\_METHOD** or directly in the **EXECUTE** method. You can use the given constants in **GCS\_HTTP\_METHOD** of interface **/IWBEPE/IF\_CP\_REQUEST\_FUNCTION**:

```
DATA: lo_function_response TYPE REF TO /iwbep/if_cp_response_function.
"Option one: Set http method on the request
lo_function_request->set_http_method( /iwbep/
if_cp_request_function=>gcs_http_method-post ).
"Option two: Directly set the http method when executing the function
lo_function_response = lo_function_request->execute( /iwbep/
if_cp_request_function=>gcs_http_method-post ).
```

### ⓘ Note

Setting the HTTP method is only allowed for OData Version 2 requests. In OData Version 4, functions must always be called with **GET** and actions with **POST**. You don't need to set the HTTP method if the function is called with **GET**.

**Step 5:** Set the **If-Match** header to provide an ETag (if needed for this request). For example, if the corresponding request header is **if-match: W/"2407"**, the required input for **SET\_IF\_MATCH** is '**2407**':

```
lo_function_request->set_if_match( '2407' ).
```

### ⓘ Note

Only Version 2 remote consumption supports the **If-Match** header.

**Step 6:** Create the Function request:

```
DATA: lo_function_response TYPE REF TO /iwbep/if_cp_response_function.
```

```
lo_function_response = lo_function_request->execute( /iwbepl  
if_cp_request_function=>gcs_http_method-post ).
```

**Step 7:** Check if the response object contains business data. This is especially useful for nullable operations, where the operation might not return data (compare to HTTP return code 204 – No Content):

```
CHECK lo_function_response->has_business_data( ) = abap_true.
```

**Step 8:** Fetch the business data from the response object (if the corresponding function does return business data):

```
DATA: ls_busi_data TYPE /iwbepl/tea_employee.  
      lo_function_response->get_business_data( IMPORTING ea_response_data =  
      ls_busi_data ).
```

### ⓘ Note

- When using the Version 4 local client proxy, the business data you enter is not converted for inbound processing. The data provider receives the data exactly how you entered it in the request. For both Version 2 and Version 4, the business data received from the (local consumption) response is also not re-converted for outbound processing.
- Using an OData Version 4 action import works the same way. Use method `CREATE_RESOURCE_FOR_ACTION` instead of `CREATE_RESOURCE_FOR_FUNCTION`.

## Constraints

- If-Match header is only supported for **remote** consumption.
- For function imports, the If-Match header is only supported for OData **Version 2** requests.
- Composable Functions are **not** supported.
- `$expand` with operations is **not** supported.
- `$select` is **not** supported for actions.
- `Return-Prefer` header is **not** supported for actions.
- System query options are **not** supported for Version 4 functions.
- If the return type of a Version 2 function is an entity type and this entity type is used by more than one entity set, the Version 2 request context only contains the (EDM) name of the first entity set with the underlying entity type. A precise mapping is currently not possible. For remote consumption, this can be handled in the proxy model by setting the correct entity set via method `/iwbepl/if_v4_med_func_imp->set_entity_set_name`

## Related Information

[Actions and Functions \[page 921\]](#)

[Bound Operation \[page 926\]](#)

## 4.2.9.3.4.5.2.2 Bound Operation

### Bound Operation

#### Overview

The starting point for invoking a bound operation is the corresponding operation resource. This resource is not created directly at the client proxy instance. It is created at the resource object the operation is bound to (for example, an entity resource or an entity list resource). You can use the resource instance to create a request instance. You can use the response instance to retrieve the response business data for the successfully executed request and can be fetched from the request instance.

#### ⓘ Note

Bound Operations are only available for OData Version 4 requests. All Operation related OData Version 2 requests must be created as Function Import requests.

#### Example

Invoke the Version 4 bound action '**IncreaseSalary**' with the non-binding (for example, input) parameters "**NewSalary=5000**" and "**Currency='EUR'**". The action is bound to one entity of the entity set "**Employees**":

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0006') /  
com.sap.gateway.default.iwbep.tea_busi.v0003.IncreaseSalary
```

```
WITH the request body  
{  
    "NewSalary" : 5000, "Currency" : "EUR"  
}
```

```
TYPES:  
BEGIN OF tys_parameter,  
    new_salary TYPE i,  
    currency TYPE c LENGTH 3,  
END OF tys_parameter,  
  
BEGIN OF tys_key,  
    id TYPE /iwbep/v4_tea_employee_id,  
END OF tys_key.  
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_action_request  TYPE REF TO /iwbep/if_cp_request_action,  
      lo_action_resource TYPE REF TO /iwbep/if_cp_resource_action,  
      lo_action_response TYPE REF TO /iwbep/if_cp_response_action,  
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity,  
      ls_parameter        TYPE tys_parameter,  
      ls_employee         TYPE /iwbep/s_v4_tea_employee,  
      ls_key              TYPE tys_key.  
ls_key = VALUE #( id = '0006' ).  
lo_entity_resource = lo_client_proxy->  
>create_resource_for_entity_set( 'EMPLOYEES')->navigate_with_key( ls_key ).  
lo_action_resource = lo_entity_resource->bind_action( 'INCREASE_SALARY' ).  
lo_action_request = lo_action_resource->create_request( ).  
ls_parameter = VALUE #( new_salary = 5000 currency = 'eur' ).  
lo_action_request->set_parameter( ls_parameter ).  
lo_action_request->set_if_match( '1234' ).
```

```

lo_action_response = lo_action_request->execute( ).
CHECK lo_action_response->has_business_data( ) = abap_true.
lo_action_response->get_business_data( IMPORTING ea_business_data =
ls_employee ).
```

## Steps

**Step 1:** Create the entity resource with the employee **Id '0006'** of the entity set '**Employees**' (with **internal** name '**EMPLOYEES**' ):

```

TYPES:
BEGIN OF tys_key,
  id TYPE /iwbp/v4_tea_employee_id,
END OF tys_key.
DATA: lo_client_proxy      TYPE REF TO /iwbp/if_cp_client_proxy,
      lo_entity_resource TYPE REF TO /iwbp/if_cp_resource_entity,
      ls_key             TYPE tys_key.
ls_key = VALUE #( id = '0006' ).
lo_entity_resource = lo_client_proxy-
>create_resource_for_entity_set( 'EMPLOYEES' )->navigate_with_key( ls_key ).
```

**Step 2:** Create the action resource instance on the instance the action is bound to (In this example, an entity of entity set '**Employees**'). '**INCREASE\_SALARY**' is the **internal** name of action '**IncreaseSalary**' :

```

DATA: lo_action_resource TYPE REF TO /iwbp/if_cp_resource_action.
lo_action_resource = lo_entity_resource->bind_action( 'INCREASE_SALARY' ).
```

**Step 3:** Create an action request instance with the action resource:

```

DATA: lo_action_request TYPE REF TO /iwbp/if_cp_request_action.
lo_action_request = lo_action_resource->create_request( ).
```

**Step 4:** Set the non-binding (for example, input) parameters of the bound action on the action request:

```

TYPES:
BEGIN OF tys_parameter,
  new_salary TYPE i,
  currency    TYPE c LENGTH 3,
END OF tys_parameter,
DATA: ls_parameter TYPE tys_parameter.
ls_parameter = VALUE #( new_salary = 5000 currency = 'eur' ).
lo_action_request->set_parameter( ls_parameter ).
```

**Step 5:** Set the **If-Match** header to provide an ETag (if needed for this request) on the action request. In this example, the corresponding request header is **if-match: W/"2407"**, the required input for **SET\_IF\_MATCH** is '**2407**' :

```
Lo_action_request->set_if_match( '2407' )
```

### ⓘ Note

Only remote consumption supports the **If-Match** header.

**Step 6:** Create the action request and get the action response instance:

```

DATA: lo_action_response TYPE REF TO /iwbp/if_cp_response_action.
lo_action_response = lo_action_request->execute( ).
```

**Step 7:** Check if the response object contains business data. This is especially useful for nullable operations, where the operation might not return data (compare to HTTP return code **204 – No Content**):

```
CHECK lo_action_response->has_business_data( ) = abap_true.
```

**Step 8:** Fetch the business data from the response object (if the corresponding action **does** return business data):

```
DATA: ls_employee TYPE /iwbep/s_v4_tea_employee.  
lo_action_response->get_business_data( IMPORTING ea_business_data = ls_employee
```

### ⓘ Note

- When using the Version 4 local client proxy, the business data you enter isn't converted for inbound processing. The data provider receives the exact data you entered in the request. The business data received from the (local consumption) response is also not re-converted for outbound processing.
- Invoking an OData Version 4 bound function works the same way. Use method **BIND\_FUNCTION** instead of **BIND\_ACTION**.

## Constraints

- If-Match header is only supported for **remote** consumption.
- For function imports, the If-Match header is only supported for OData **Version 2** requests.
- Composable Functions are **not** supported.
- **\$expand** with operations is **not** supported.
- **\$select** is **not** supported for actions.
- **Return-Prefer** header is **not** supported for actions.
- System query options are **not** supported for Version 4 functions.
- If the return type of a Version 2 function is an entity type and this entity type is used by more than one entity set, the Version 2 request context only contains the (EDM) name of the first entity set with the underlying entity type. A precise mapping is currently not possible. For remote consumption, this can be handled in the proxy model by setting the correct entity set via method `. /iwbep/if_v4_med_func_imp->set_entity_set_name`

## Related Information

[Actions and Functions \[page 921\]](#)

[Import Operation \[page 923\]](#)

## 4.2.9.3.4.6 OData Requests

Learn how to use OData Requests.

OData is a standardized protocol built over existing HTTP and REST protocols supporting CRUD (Create, Read, Update, Delete) operations for creating and consuming data APIs. There are various types of requests (including **Create**, **Update**, **Read**, and more) and system query options you can use for your particular use case.

### [OData Request Terms \[page 930\]](#)

An overview of some OData Request terminology.

### [OData Request as Batch Including Changesets \[page 930\]](#)

Create an \$batch request, including changesets in the Client Proxy instance.

### [OData Request: Create Entity \[page 939\]](#)

Create an entity in the Client Proxy instance with insert entity request.

### [OData Request: Update Entity \[page 943\]](#)

Create an OData request to update an entity in the Client Proxy instance.

### [OData Request: Read Entity \[page 946\]](#)

To create an OData request to read an entity in the Client Proxy instance.

### [OData Request: Read Optional Entity \[page 949\]](#)

Create an OData request to read an optional entity in the Client Proxy instance.

### [OData Request: Delete Entity \[page 951\]](#)

Create an OData request to delete an entity in the Client Proxy instance.

### [OData Request: Read Entity List \[page 953\]](#)

Create an OData request to read an entity list (entity collection) in the Client Proxy instance.

### [OData Request: Update Entity List \[page 955\]](#)

Create an OData request to update an entity list in the Client Proxy instance.

### [OData Request: Custom Query Option \[page 959\]](#)

Create an OData request using a custom query option that isn't one of the OData-defined system query options in the Client Proxy instance.

### [OData Request: Deep Create \[page 961\]](#)

Create an OData request to execute a "deep create" (deep insert) in the Client Proxy instance.

### [OData Request: Delta Link Query Option \[page 965\]](#)

Create an OData request with \$delta token query option in the Client Proxy instance.

### [OData Request: Using Navigation \[page 968\]](#)

Create an OData request using a navigation in the Client Proxy instance.

### [OData Request: Including a Nextlink \[page 970\]](#)

Create an OData request to read an entity list (entity collection) with a next link in the Client Proxy instance.

### [OData Request System Query Options \[page 972\]](#)

Learn how to use OData system query options. System query options include:

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Outbound Communication \[page 897\]](#)

[OData Request System Query Options \[page 972\]](#)

### 4.2.9.3.4.6.1 OData Request Terms

An overview of some OData Request terminology.

Here's a list of some common terms you'll see when creating OData Requests.

Term	Definition
Request	OData Protocol enables clients to make requests and retrieve responses from the OData service.
Entity	An entity is an instance of the <b>EntityType</b> element (for example, Customer or Employee) that is a structured record with a key that uniquely identifies it.
Entity Key	Each entity has a key that uniquely identifies it (for example, CustomerId or EmployeeId). The key is defined by a subset of the entity type properties.
Entity Type	An entity type is a collection or category that a particular entity belongs to.
Entity Set	An entity set is a group of all entities (for example, the EntitySet Customers is a set of Customer instances) for a particular entity type at any point in time.
Property	An entity type has one or more properties that define the entity type's structure. Each property has a name and one or more values.

### 4.2.9.3.4.6.2 OData Request as Batch Including Changesets

Create an \$batch request, including changesets in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A **batch** request is a single standard HTTP request that includes multiple application requests. Within that main HTTP request, each of the parts contains a nested HTTP request. \$batch request is useful to group

multiple requests and sends a batch to the data service in a single HTTP request. This section explains how a Batch Request works using ChangeSet syntax to logically group requests into a single value in a batch.

## OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

**Batch** requests group multiple individual requests into a single HTTP request payload. An individual request in a **batch** request is one of the following:

- Metadata request
- Data request
- Data modification request
- Action invocation request
- Function invocation request

**Batch** requests are sent as a single HTTP POST request to the **batch** endpoint of a service in the URL `$batch` that is relative to the service root.

Individual requests in a batch request are evaluated according to the same semantics that are used when the request isn't part of a **batch** request.

In the Multipart format, data modification or action invocation requests can be grouped as part of an atomic change set. Operations outside the change set are executed sequentially. Operations within the change set are executed in any order.

## Example Requests

### Version 4

Get all entities of entity set “**EMPLOYEES**” with Id = ‘1’ and execute the Action Import “**ChangeTeamBudgetByID**”:

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/$batch
```

With request body

```
--batch
Content-Type: application/http
Content-Transfer-Encoding: binary
GET EMPLOYEES?$filter=ID%20eq%20%271%27 HTTP/1.1
--batch
Content-Type: multipart/mixed;boundary=change_set_1
--change_set_1
Content-Type: application/http
Content-Transfer-Encoding: binary
Content-ID: 1
POST ChangeTeamBudgetByID HTTP/1.1
Content-Type: application/json
{
    "TeamID" : "TEAM_01",
    "Budget" : 700.00
}
--change_set_1--
--batch--
```

## Version 2

Update two entities of entity set '**Conversions**'

```
POST /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/$batch
```

With request body

```
--batch
Content-Type: multipart/mixed; boundary=changeset
--changeset Content-Type: application/http
Content-Transfer-Encoding: binary
PUT Conversions(Id='0001') HTTP/1.1
Content-Type: application/json
{
    "Id" : "1",
    "price_1" : "-180.59",
    "currency_1" : "EUR",
    "price_2" : "1.706",
    "currency_2" : "KWD",
    "amount_1" : "1.1235",
    "unit_1" : "D",
    "amount_2" : "1.123",
    "unit_2" : "CMS",
    "oSQL_Where_Cl" : "",
    "Is_Boolean" : true
}
--changeset--
--batch
Content-Type: multipart/mixed; boundary=changeset
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
PUT Conversions(Id='0001') HTTP/1.1
Content-Type: application/json
{
    "Id" : "1",
    "price_1" : "-000.00",
    "currency_1" : "EUR",
    "price_2" : "1.706",
    "currency_2" : "KWD",
    "amount_1" : "1.1235",
    "unit_1" : "D",
    "amount_2" : "1.123",
    "unit_2" : "CMS",
    "oSQL_Where_Cl" : "",
    "Is_Boolean" : true
}
--changeset--
--batch--
```

## Batch with Changesets Request

### Overview

#### Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **BATCH** request on an entity.

The starting point for a **\$batch** request is the Client Proxy instance. It's possible to create a **\$batch** request that you can use to create a changeset request.

## Example

Get all entities of entity set “**Employees**” and create a new entity in entity set “**Teams**”:

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/$batch
```

With request body

```
--batch
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Employees HTTP/1.1
--batch
Content-Type: multipart/mixed;boundary=change_set_1
--change_set_1
Content-Type: application/http
Content-Transfer-Encoding: binary
Content-ID: 1
POST Teams HTTP/1.1
Content-Type: application/json
{
    "Id" : "TEAM_04",
    "Name" : "Test Team"
}
--change_set_1--
--batch--
```

```
DATA:      lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
          lo_read_request   TYPE REF TO /iwbep/if_cp_request_read_list,
          lo_read_response   TYPE REF TO /iwbep/if_cp_response_read_lst,
          lo_create_request  TYPE REF TO /iwbep/if_cp_request_create,
          lo_create_response  TYPE REF TO /iwbep/if_cp_response_create,
          lo_batch_request    TYPE REF TO /iwbep/if_cp_request_batch,
          lo_changeset_request  TYPE REF TO /iwbep/if_cp_request_changest,
          lt_employee         TYPE STANDARD TABLE OF /iwbep/
s_v4_tea_employee,ls_team
          TYPE /iwbep/s_v4_tea_team.
lo_batch_request = lo_client_proxy->create_request_for_batch( ).
lo_changeset_request = lo_batch_request->create_request_for_changest( ).
lo_changeset_request->add_request( lo_create_request ).

lo_batch_request->add_request( lo_read_request ).
lo_batch_request->add_request( lo_changeset_request ).
lo_batch_request->execute( ).
lo_batch_request->check_execution( ).
lo_changeset_request->check_excecution( ).
lo_read_request->check_execution( ).
lo_create_request->check_execution( ).
lo_read_response = lo_read_request->get_response( ).
lo_read_response->get_business_data( IMPORTING et_business_data =
lt_employee ).

lo_create_response = lo_create_request->get_response( ).
lo_create_response->get_business_data( IMPORTING es_business_data = ls_team ).
```

## Steps

**Step 1:** Create the **\$batch** request at the client proxy instance:

```
DATA:      lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
```

```
lo_batch_request TYPE REF TO /iwbp/if_cp_request_batch.  
lo_batch_request = lo_client_proxy->create_request_for_batch( ).
```

**Step 2:** Create the changeset request using the **\$batch** request instance:

```
DATA:      lo_batch_request      TYPE REF TO /iwbp/if_cp_request_batch,  
          lo_changeset_request TYPE REF TO /iwbp/if_cp_request_changset.  
          lo_changeset_request = lo_batch_request-  
>create_request_for_changset( ).
```

**Step 3:** Add the **CREATE** request to the changeset request:

```
DATA:      lo_create_request      TYPE REF TO /iwbp/if_cp_request_create,  
          lo_changeset_request TYPE REF TO /iwbp/if_cp_request_changset.  
          lo_changeset_request->add_request( lo_create_request ).
```

**Step 4:** Add both the **READ** and the **change\_set** request into the **\$batch** request. Run the **\$batch** request:

```
DATA:      lo_read_request      TYPE REF TO /iwbp/if_cp_request_read_list,  
          lo_batch_request      TYPE REF TO /iwbp/if_cp_request_batch,  
          lo_changeset_request TYPE REF TO /iwbp/if_cp_request_changset.  
          lo_batch_request->add_request( lo_read_request ).  
          lo_batch_request->add_request( lo_changeset_request ).  
          lo_batch_request->execute( ).
```

**Step 5:** Check that the four requests ran successfully. (optional):

```
DATA:      lo_read_request      TYPE REF TO /iwbp/if_cp_request_read_list,  
          lo_create_request      TYPE REF TO /iwbp/if_cp_request_create,  
          lo_batch_request      TYPE REF TO /iwbp/if_cp_request_batch,  
          lo_changeset_request TYPE REF TO /iwbp/if_cp_request_changset.  
          lo_batch_request->check_execution( ).  
          lo_changeset_request->check_execution( ).  
          lo_read_request->check_execution( ).  
          lo_create_request->check_execution( ).
```

## ⓘ Note

If the exception didn't run successfully, **CHECK\_EXECUTION** raises an exception. See [Handling Exceptions \[page 916\]](#) for more information.

**Step 6:** Get the **READ** response instance using the **READ** request instance and use it to fetch the corresponding business data:

```
DATA:      lo_read_request      TYPE REF TO /iwbp/if_cp_request_read_list,  
          lo_read_response      TYPE REF TO /iwbp/if_cp_response_read_lst,  
          lt_employee  
          TYPE STANDARD TABLE OF /iwbp/s_v4_tea_employee.  
          lo_read_response = lo_read_request->get_response( ).  
          lo_read_response->get_business_data( IMPORTING et_business_data =  
          lt_employee ).
```

**Step 7:** Get the **CREATE** response instance using the **CREATE** request instance and use it to fetch the corresponding business data:

```
DATA:      lo_create_request    TYPE REF TO /iwbp/if_cp_request_create,  
          lo_create_response    TYPE REF TO /iwbp/if_cp_response_create,  
          ls_team  
          TYPE /iwbp/s_v4_tea_team.  
          lo_create_response = lo_create_request->get_response( ).
```

```
lo_create_response->get_business_data( IMPORTING es_business_data =  
ls_team ).
```

### ⓘ Note

When you're using the Version 4 local client proxy, the business data you receive from the local consumption response isn't converted again for outbound processing.

## Related Information

[OData Request Terms \[page 930\]](#)

### 4.2.9.3.4.6.2.1 Content ID Referencing

Create an OData \$batch request using Content ID Referencing in the Client Proxy instance.

## OData Specification

### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

Multipurpose Internet Mail Extensions (MIME) is set of extensions that redefines and expands support for various types of content in email messages. Each MIME part that represents a request in a Change Set can include a Content-ID MIME header. See [\[RFC2045 Section 7: Content-ID Header Field\]](#) for more information.

If a MIME part defines an **InsertEntity** request and includes a **Content-ID** header, the new entity is defined by the **InsertEntity** request. You can reference the new entity in future requests in the Change Set using the "**\$<Content-ID value of your previous request>**" token in place of a Resource Path that identifies the new resource. The token acts as an alias for the Resource Path that identifies the new entity. Requests in different Change Sets can't reference one another, even if they are in the same Batch.

### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

Each request in a batch request can have an assigned request identifier. The request identifier:

- is case-sensitive.
- Must be unique within the batch request.
- Must satisfy the rule request-id in [\[OData-ABNF\]](#).

The body part contents that represent a change set must be a multipart document with one body part for each operation in the change set. See [\[RFC2046\]](#). Each body part that represents an operation in the change set must specify a **Content-ID** header with a unique request identifier in the batch request.

## Example Requests

### Version 4

Create new entity in entity set “**TEAMS**” and update it afterwards:

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/$batch
```

With request body

```
--batch
content-type: multipart/mixed; boundary=changeset
--changeset
content-type: application/http
content-id: 111
POST TEAMS HTTP/1.1
Content-Type: application/json
{
  "Name": "Unit Test Task Force",
  "MANAGER_ID": "3",
  "BudgetCurrency": "JPY",
  "Budget": 111100
}
--changeset
content-type: application/http
content-id: 222
PATCH $111 HTTP/1.1
Content-Type: application/json
{
  "MEMBER_COUNT": 66
}
--changeset--
--batch-
```

### Version 2

Create a new entity in entity set “**Decimals**”, then read it:

```
POST /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/$batch
```

With request body

```
--batch
Content-Type: multipart/mixed; boundary=changeset
--changeset
Content-Type: application/http
Content-Transfer-Encoding: binary
POST Decimals HTTP/1.1
Content-Type: application/json
Accept: application/json
Content-Id: 1
{
  "Id": 1,
  "Name": "Decimal 10",
  "Decimal1": "1.1000000000"
}
--changeset--
--batch
Content-Type: application/http
Content-Transfer-Encoding: binary
Accept: application/json
GET $1?$format=json HTTP/1.1
--batch--
```

## Create a \$batch request using Content ID Referencing

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to create an OData \$batch request using Content ID Referencing.

The starting point for a using Content ID Referencing is a create request instance and create an entity resource with a Content ID reference.

### Example

Create a new entity in entity set “**Teams**”, then update the member count:

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/$batch
```

With request body

```
--batch
content-type: multipart/mixed; boundary=changeset
--changeset
content-type: application/http
content-id: 111
POST Teams HTTP/1.1
Content-Type: application/json
{
  "Name": "Unit Test Task Force"
}
--changeset
content-type: application/http
content-id: 222
PATCH $111 HTTP/1.1
Content-Type: application/json
{
  "MemberCount": 42
}
--changeset--
--batch--
```

```
DATA:      lo_create_request      TYPE REF TO /iwbep/if_cp_request_create,
         lo_batch_request        TYPE REF TO /iwbep/if_cp_request_batch,
         lo_changeset_request    TYPE REF TO /iwbep/if_cp_request_changeset,
         lo_update_request       TYPE REF TO /iwbep/if_cp_request_update,
         lo_entity_resource_w_cir TYPE REF TO /iwbep/if_cp_resource_entity,
         ls_patch_data           TYPE /iwbep/s_v4_tea_team.
         lo_entity_resource_w_cir = lo_create_request-
>create_res_w_content_id_ref( ).
         lo_update_request
= lo_entity_resource_w_cir->create_request_for_update( /iwbep/
if_cp_request_update=>gcs_update_semantic-patch ).
         ls_patch_data = VALUE #( member_count = 42 ).
         lo_update_request->set_business_data( is_business_data = ls_patch_data
It_provided_property = VALUE #( ( `MEMBER_COUNT` ) ) ).
```

lo\_changeset\_request->add\_request( lo\_create\_request ).  
lo\_changeset\_request->add\_request( lo\_update\_request ).  
lo\_batch\_request->add\_request( lo\_changeset\_request ).  
lo\_batch\_request->execute( ).

## Steps

**Step 1:** Create the entity resource with a Content ID reference in the create request instance:

```
DATA:      lo_create_request TYPE REF TO /iwbep/if_cp_request_create,
          lo_entity_resource_w_cir TYPE REF TO /iwbep/if_cp_resource_entity.
          lo_entity_resource_w_cir = lo_create_request-
>create_res_w_content_id_ref( ).
```

**Step 2:** Create the update request instance at the entity resource instance:

```
DATA:      lo_update_request TYPE REF TO /iwbep/if_cp_request_update,
          lo_entity_resource_w_cir TYPE REF TO /iwbep/if_cp_resource_entity.
          lo_update_request
= lo_entity_resource_w_cir->create_request_for_update( /iwbep/
if_cp_request_update=>gcs_update_semantic-patch ).
```

**Step 3:** Define the corresponding update data and set it into the update request:

```
DATA:      lo_update_request TYPE REF TO /iwbep/if_cp_request_update,
          ls_patch_data TYPE /iwbep/s_v4_tea_team.
          ls_patch_data = value #( member_count = 42 ).
          lo_update_request->set_business_data( is_business_data = ls_patch_data
          It_provided_property = VALUE #( ( `MEMBER_COUNT` ) ) ).
```

**Step 4:** Add the **CREATE** and **UPDATE** requests into the changeset request. Then add the changeset request into the **\$batch** request:

```
DATA:      lo_create_request      TYPE REF TO /iwbep/if_cp_request_create,
          lo_batch_request        TYPE REF TO /iwbep/if_cp_request_batch,
          lo_changeset_request    TYPE REF TO /iwbep/if_cp_request_changeset,
          lo_update_request       TYPE REF TO /iwbep/if_cp_request_update.

          lo_changeset_request->add_request( lo_create_request ).
          lo_changeset_request->add_request( lo_update_request ).
          lo_batch_request->add_request( lo_changeset_request ).
```

**Step 5:** Run the **\$batch** request:

```
DATA:      lo_batch_request TYPE REF TO /iwbep/if_cp_request_batch.
          lo_batch_request->execute( ).
```

## Constraints

- You can't use Content ID Referencing with navigation (for example, **POST \$1/NavigationProperty**).

## Related Information

[Request Instance \[page 913\]](#)

[Resource Instance \[page 910\]](#)

[OData Request as Batch Including Changesets \[page 930\]](#)

## 4.2.9.3.4.6.3 OData Request: Create Entity

Create an entity in the Client Proxy instance with insert entity request.

### OData Specification

#### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

The **InsertEntity** request enables a new EntityType instance with new related entities you add to an EntitySet.

#### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

To create an entity in a collection, the client sends a **POST** request to the collection's URL. The **POST** body MUST contain a single valid entity representation.

### Example Requests

#### Version 4

Create the employee with id '**007**' in the entity set "**Employees**"

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees
```

With request body

```
{
  "Id" : "007",
  "Name" : "James Bond",
  "Gun" : "Walther PPK",
  "Car" : "Aston Martin DB5"
}
```

#### Version 2

Create the employee with id '**007**' in the entity set "**Employees**"

```
POST /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees
```

With request body

```
{
  "Id" : "007",
  "Name" : "James Bond",
  "Gun" : "Walther PPK",
  "Car" : "Aston Martin DB5"
}
```

## Create Entity Request

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **CREATE** request on an entity..

The starting point for a **POST** request on an entity list is the Client Proxy instance. It's possible to create an entity list resource, which can then be used to create a create request.

Running the create request provides the create response instance, which can return the business data for the **CREATE** entity request.

### Example

Create the employee with key id = '007' of the Version 4 entity set '**Employees**':

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees
```

With request body

```
{
    "Id" : "007",
    "Name" : "James Bond",
    "Age" : 38,
    "Entrydate" : "1962-10-05",
    "ManagerId" : "001",
    "RoomId" : "1",
    "TeamId" : "TEAM_BRITANIA",
    "IsManager" : false,
    "Status" : "Available"
}
TYPES: BEGIN OF tys_entity_data,
    Id          TYPE i,
    Name        TYPE string,
    age         TYPE i,
    entry_date TYPE dats,
    manager_id TYPE i,
    room_id    TYPE i,
    team_id    TYPE string,
    is_manager  TYPE abap_bool,
    status      TYPE string,
END OF tys_entity_data.

DATA: lo_client_proxy          TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_create_request      TYPE REF TO /iwbep/if_cp_request_create,
      lo_create_response     TYPE REF TO /iwbep/if_cp_response_create,
      lo_entity_list_resource TYPE REF TO /iwbep/if_cp_resource_list,
      ls_entity_data         TYPE tys_entity_data, ls_employee
      TYPE /iwbep/s_v4_tea_employee.
      lo_entity_list_resource = lo_client_proxy-
>create_resource_for_entity_set( 'EMPLOYEES' ).
      lo_create_request = lo_entity_list_resource-
>create_request_for_create( ).  

      ls_entity_data = VALUE #( id = 007
```

```

        name = 'James Bond'
        age = 38
        entry_date = '19621005'
        manager_id = 001
        room_id = 1
        team_id = 'team_britania'
        is_manager = abap_false
        status = 'Available'      .
    lo_create_request->set_business_data( is_business_data = ls_entity_data
        It_provided_property = VALUE #( ( |ID| )
            ( |NAME| )
            ( |AGE| )
            ( |ENTRY_DATE| )
            ( |MANAGER_ID| )
            ( |ROOM_ID| )
            ( |TEAM_ID| )
            ( |IS_MANAGER| )
            ( |STATUS| ) .
    lo_create_response = lo_create_request->execute( ).
    lo_create_response->get_business_data( IMPORTING es_business_data =
ls_employee ).
```

## Steps

**Step 1:** Create the entity list resource for entity set '**Employees**' (with **internal** name '**EMPLOYEES**' ):

```

DATA:      lo_client_proxy          TYPE REF TO /iwbep/if_cp_client_proxy,
          lo_entity_list_resource  TYPE REF TO /iwbep/if_cp_resource_list.
          lo_entity_list_resource = lo_client_proxy-
>create_resource_for_entity_set( 'EMPLOYEES' ).
```

**Step 2:** Run the create request instance on the entity list resource.

```

DATA:      lo_update_request         TYPE REF TO /iwbep/if_cp_request_create,
          lo_entity_list_resource  TYPE REF TO /iwbep/if_cp_resource_list.
          lo_create_request = lo_entity_list_resource-
>create_request_for_create( ).
```

**Step 3:** Set the business data for your create request. In our example, the create request properties are:

Properties	Internal Names
Id	<b>ID</b>
Name	<b>NAME</b>
Age	<b>AGE</b>
EntryDate	<b>ENTRY_DATE</b>
ManagerId	<b>MANAGER_ID</b>
RoomId	<b>ROOM_ID</b>
TeamId	<b>TEAM_ID</b>
Status	<b>STATUS</b>
IsManager	<b>IS_MANAGER</b>

Define the values and set the business data into the create request:

```

TYPES:      BEGIN OF tys_entity_data,
            Id          TYPE i,
            Name        TYPE string,
```

```

        age      TYPE i,
        entry_date  TYPE dats,
        manager_id  TYPE i,
        room_id     TYPE i,
        team_id     TYPE string,
        is_manager  TYPE abap_bool,
        status      TYPE string,
        END OF tys_entity_data.
DATA:    lo_update_request  TYPE REF TO /iwbep/if_cp_request_create,
        ls_entity_data    TYPE tys_entity_data,
        ls_employee       TYPE /iwbep/s_v4_tea_employee.
        ls_entity_data = VALUE #( id = 007
            name = 'James Bond'
            age = 38
            entry_date = '19621005'
            manager_id = 001
            room_id = 1
            team_id = 'team_britania'
            is_manager = abap_false
            status = 'Available' ).
        lo_create_request->set_business_data( is_business_data = ls_entity_data
            it_provided_property = VALUE #( ( |ID| )
                ( |NAME| )
                ( |AGE| )
                ( |ENTRY_DATE| )
                ( |MANAGER_ID| )
                ( |ROOM_ID| )
                ( |TEAM_ID| )
                ( |IS_MANAGER| )
                ( |STATUS| ) ).
```

#### ⓘ Note

(Optional) Provide the properties when calling **SET\_BUSINESS\_DATA**. If you provide properties, only the properties you set are considered. If you don't provide properties, all properties are considered for the **CREATE** request.

#### ⓘ Note

If the entity has value control or VCS properties, these properties **should** also be part of the provided data container (**ls\_entity\_data** in this example). If the value control/VCS properties are not provided, the behavior (for example: conversion exits) is undefined and can cause unexpected side-effects.

**Step 4:** Run the **create** request and get the create response instance:

```

DATA:    lo_update_request  TYPE REF TO /iwbep/if_cp_request_create,
        lo_update_response TYPE REF TO /iwbep/if_cp_response_create.
        lo_create_response = lo_create_request->execute( ).
```

**Step 5:** You fetch the business data from the response object:

```

DATA:    lo_update_response  TYPE REF TO /iwbep/
        if_cp_response_create,ls_employee
        TYPE /iwbep/s_v4_tea_employee.
        lo_create_response->get_business_data( IMPORTING es_business_data =
        ls_employee ).
```

#### ⓘ Note

When you're using the Version 4 local client proxy, the business data you receive from the local consumption response isn't converted again for outbound processing.

## Related Information

[OData Request Terms \[page 930\]](#)

### 4.2.9.3.4.6.4 OData Request: Update Entity

Create an OData request to update an entity in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

An **UpdateEntity** Request is used by a client to update an existing AtomPub Entry Resource, as specified in [RFC5023](#), that maps to an EntityType instance in the abstract data model.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

To update an individual entity, the client makes a **PATCH** or **PUT** request to a URL that identifies the entity. Services can be restricted to only request updates addressing the edit URL of the entity.

## Example Requests

### Version 4

Update the employee with id '007' of entity set "Employees"

```
PATCH /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
```

With request body

```
{  
    "Car" : "Aston Martin DB5"  
}
```

### Version 2

Update the employee with id '007' of entity set "Employees":

```
PUT /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees('007')
```

With request body

```
{ "Id" : "007",
```

```

        "Name" : "James Bond",
        "Gun" : "Walther PPK",
        "Car" : "Aston Martin DB5"
    }

```

## Update Entity Request

### Overview

#### Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **UPDATE** request on an entity.

The starting point for a **PUT/PATCH** entity request is the Client Proxy instance. You can create an **entity resource** based on an **entity list resource**, and use it to create an update request.

Running the update request provides the update response, which can return the business data of the **UPDATE** entity request.

### Example

Update the employee with key “id = ‘007’” of the Version 4 entity set ‘**Employees**’:

```
PATCH /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
```

With request body

```

{
    "Age" : 38,
    "IsManager" : true,
    "Status" : "Available"
}

TYPES: BEGIN OF tys_patch_data,
types: BEGIN OF tys_patch_data,
age      TYPE i,
is_manager TYPE abap_bool,
status    TYPE string,
END OF tys_patch_data.
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
lo_update_request   TYPE REF TO /iwbep/if_cp_request_update,
lo_update_response   TYPE REF TO /iwbep/if_cp_response_update,
lo_entity_resource   TYPE REF TO /iwbep/if_cp_resource_entity,
ls_patch_data         TYPE tys_patch_data,
ls_employee           TYPE /iwbep/s_v4_tea_employee.
lo_entity_resource

= lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES')-
>navigate_with_key(value /iwbep/if_v4_tea_busi_types=>ty_s_employee_key( id =
'007' ) .
    lo_update_request = lo_entity_resource->create_request_for_update( /
iwbep/if_cp_request_update=>gcs_update_semantic-patch ) .
    ls_patch_data = VALUE #( age = 38
        is_manager = abap_true
        status = 'Available' ).
    lo_update_request->set_business_data( is_business_data = ls_patch_data
it_provided_property = VALUE #( ( |AGE| )
        ( |IS_MANAGER| ) )

```

```

        ( |STATUS| ) .

lo_update_request->set_if_match( '1234' ) .
lo_update_response = lo_update_request->execute( ) .
lo_update_response->get_business_data( IMPORTING es_business_data =
ls_employee ) .

```

## Steps

**Step 1:** Create the entity resource for the employee with the following:

- key “Id=‘007’” in the entity set ‘**Employees**’ (**internal** name ‘**EMPLOYEES**’)
- type “**ty\_s\_employee\_key**” in the interface **/iwbp/if\_v4\_tea\_busi\_types**”

```

DATA      lo_client_proxy      TYPE REF TO /iwbp/if_cp_client_proxy,
          lo_entity_resource  TYPE REF TO /iwbp/if_cp_resource_entity.
          lo_entity_resource
= lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES' )-
>navigate_with_key(value /iwbp/if_v4_tea_busi_types=>ty_s_employee_key( id =
'007' ) .

```

**Step 2:** Create the update request instance on the entity resource. You need to provide the correct semantic (either **PATCH** or **PUT**). **PUT** creates a full new resource representation. If some attributes are not provided, these attributes should be removed (set to null) or set to their default value. **PATCH** also updates a resource, but unlike **PUT**, it applies a delta rather than replacing the entire resource. The **PATCH** payload is a different content-type than the entity that it is modifying. Instead of being a full resource, it is a resource that describes modifications that be made to a resource.

In our example, we use **PATCH** semantic. You can use the constants in structure **gcs\_update\_semantic** of interface **/iwbp/if\_cp\_request\_update** as input:

```

DATA:      lo_update_request  TYPE REF TO /iwbp/if_cp_request_update,
          lo_entity_resource  TYPE REF TO /iwbp/if_cp_resource_entity.
          lo_update_request = lo_entity_resource->create_request_for_update( /
iwbp/if_cp_request_update=>gcs_update_semantic-patch ) .

```

**Step 3:** Set the business (for example, update) data for your update request. In our example, we want to update three properties. We define the new values and set the business data into the update request:

Properties	Internal Names
Age	<b>AGE</b>
Status	<b>STATUS</b>
IsManager	<b>IS_MANAGER</b>

```

TYPES:   BEGIN OF tys_patch_data,
         age           TYPE i,
         is_manager    TYPE abap_bool,
         status        TYPE string,
         END OF tys_patch_data.
DATA:     lo_update_request TYPE REF TO /iwbp/if_cp_request_update,
          ls_patch_data    TYPE tys_patch_data.
          ls_patch_data = VALUE #( age = 38
          is_manager = abap_true
          status = 'Available' ).
          lo_update_request->set_business_data( is_business_data = ls_patch_data
          it_provided_property = VALUE #( ( |AGE| )
                                         ( |IS_MANAGER| )
                                         ( |STATUS| ) ) .

```

### Note

Providing the properties is required when calling `SET_BUSINESS_DATA` for a `PATCH` request.

**Step 4:** Set the `If-Match` header to provide a ETag (if needed for this request). For example, if the corresponding request header is `if-match: W/"1234"`, the required input for `SET_IF_MATCH` is '`1234`':

```
DATA:      lo_update_request TYPE REF TO /iwbep/if_cp_request_update.  
          lo_update_request->set_if_match( '1234' ).
```

**Step 5:** Run the update request and get the update response instance:

```
DATA:      lo_update_request  TYPE REF TO /iwbep/if_cp_request_update,  
          lo_update_response TYPE REF TO /iwbep/if_cp_response_update.  
          lo_update_response = lo_update_request->execute( ).
```

**Step 6:** Fetch the business data from the response object:

```
DATA:      lo_update_response TYPE REF TO /iwbep/if_cp_response_update,  
          ls_employee           TYPE /iwbep/s_v4_tea_employee.  
          lo_update_response->get_business_data( IMPORTING es_business_data =  
          ls_employee ).
```

### Note

When you're using the Version 4 local client proxy, the business data you receive from the local consumption response isn't converted again for outbound processing.

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.5 OData Request: Read Entity

To create an OData request to read an entity in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A client uses a RetrieveEntity request to retrieve an AtomPub Entry Resource. See [RFC5023: Atom Publishing Protocol](#) for more information on the Atom Publishing Protocol.

## OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

OData services support requests for data using HTTP GET requests. The URL path includes the request target (for example, the collection of entities, entity, navigation property, structural property, or operation).

## Example Requests

### Version 4

Get the employee with id '007' from the "Employees" entity set.

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
```

### Version 2

Get the employee with id '007' from the "Employees" entity set.

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees('007')
```

## OData Request: Read Entity

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **READ** request on an entity.

The starting point for a **GET** entity request is the Client Proxy instance. You can create an entity resource based on an entity list resource, which can use create an entity resource based on an entity list resource. Use the entity resource to create a read request.

Running the read request provides a response, which can return the business data of the **READ** entity request.

### Example

From the Version 4 entity set 'Employees', fetch the employee with key id '007':

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
```

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_read_request    TYPE REF TO /iwbep/if_cp_request_read,
      lo_read_response   TYPE REF TO /iwbep/if_cp_response_read,
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity,
      ls_employee        TYPE /iwbep/s_v4_tea_employee.
      lo_entity_resource = lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES' )-
>navigate_with_key( VALUE /iwbep/if_v4_tea_busi_types=>ty_s_employee_key( id =
'007' ) .
      lo_read_request = lo_entity_resource->create_request( ).
```

```
lo_read_response = lo_read_request->execute( ).  
lo_read_response->get_business_data( IMPORTING es_business_data =  
ls_employee ).
```

## Steps

To run an OData request to read an entity:

**Step 1:** Create the entity resource for the employee with the key `id='007'` from the entity set 'Employees' (with internal name '`EMPLOYEES'`). Use type `ty_s_employee_key` in interface `/iwbep/if_v4_tea_busi_types` that includes the key property `id`:

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.  
      lo_entity_resource  
= lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES' )-  
>navigate_with_key( VALUE /iwbep/if_v4_tea_busi_types=>ty_s_employee_key( id =  
'007' ) ).
```

**Step 2:** Create the read request instance on the entity resource:

```
DATA: lo_read_request      TYPE REF TO /iwbep/if_cp_request_read,  
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.  
      lo_read_request = lo_entity_resource->create_request( ).
```

**Step 3:** Execute the read request and get the read response instance:

```
DATA: lo_read_request      TYPE REF TO /iwbep/if_cp_request_read,  
      lo_read_response TYPE REF TO /iwbep/if_cp_response_read.  
      lo_read_response = lo_read_request->execute( ).
```

**Step 4:** Fetch the business data from the response object:

```
DATA: lo_read_response TYPE REF TO /iwbep/if_cp_response_read,  
      ls_employee        TYPE /iwbep/s_v4_tea_employee.  
      lo_read_response->get_business_data( IMPORTING es_business_data =  
ls_employee ).
```

### ⓘ Note

When you're using the Version 4 local client proxy, the business data you receive from the local consumption response isn't converted again for outbound processing.

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.6 OData Request: Read Optional Entity

Create an OData request to read an optional entity in the Client Proxy instance.

An optional entity is accessed using navigation with cardinality (for example, the measure of the number of elements in a set) zero-to-one (for example, the accessed entity has either zero or one entry).

### OData Specification

#### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

The client uses a **RetrieveEntity** Request to retrieve an AtomPub Entry Resource, as specified in [RFC5023](#), and potentially related entities that map to EntityType instances.

#### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

OData services support requests for data with HTTP GET requests. The URL path specifies the target of the request (for example, the collection of entities, entity, navigation property, structural property, or operation).

### Example Requests

#### Version 4

Get the manager of the Employee with Id '**0004**' via zero-to-one navigation property '**Employee2Manager**':

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0004')/Employee2Manager
```

#### Version 2

Get the technical info of the Team '**TEAM\_02**' via zero-to-one navigation property "**Technical\_Info**"

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Teams('TEAM_02')/Technical_Info
```

### Read Optional Entity Request

#### Overview

##### Note

As the coding is independent of the OData version, we're presenting a general example on how to read a an optional entity .

The starting point for a **\$read** optional entity request is an optional entity in the Client Proxy instance. It's possible to create a **\$read** request with a corresponding zero-to-one navigation.

On the optional entity resource, it's possible to create an optional read request instance.

Running the zero-to-one read request provides the zero-to-one read response, which can return the business data of the optional READ entity request.

## Example

Get the manager of the Employee with Id '**0004**' via zero-to-one navigation property '**Employee2Manager**':

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0004')/Employee2Manager
```

## Steps

```
DATA: lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity01,
      lo_read_request     TYPE REF TO /iwbep/if_cp_request_read_01,
      lo_read_response    TYPE REF TO /iwbep/if_cp_response_read_01,
      lt_manager          TYPE STANDARD TABLE OF /iwbep/s_v4_tea_manager.
lo_read_request = lo_entity_resource->create_request_for_read( ).
lo_read_response = lo_read_request->execute( ).
lo_read_response->get_business_data( IMPORTING et_business_data = lt_manager ).
```

**Step 1:** Create the read request instance using the entity resource instance:

```
DATA: lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity01,
      lo_read_request     TYPE REF TO /iwbep/if_cp_request_read_01.
lo_read_request = lo_entity_resource->create_request_for_read( ).
```

### ⓘ Note

See [OData Request: Using Navigation \[page 968\]](#) for more information about how to create an optional entity resource instance.

### ⓘ Note

Instead of a **READ** request, you can also set up an **UPDATE** or **DELETE** request for the optional entity.

**Step 2:** Execute the **read** request and fetch the corresponding read response instance:

```
DATA: lo_read_request  TYPE REF TO /iwbep/if_cp_request_read_01,
      lo_read_response TYPE REF TO /iwbep/if_cp_response_read_01.
lo_read_response = lo_read_request->execute( ).
```

**Step 3:** Retrieve the business data from the read response instance:

```
DATA: lo_read_response TYPE REF TO /iwbep/if_cp_response_read_01,
      lt_manager        TYPE STANDARD TABLE OF /iwbep/s_v4_tea_manager.
lo_read_response->get_business_data( IMPORTING et_business_data = lt_manager ).
```

### ⓘ Note

The zero-to-one read request response is always a table that contains either zero or one entry.

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Using Navigation \[page 968\]](#)

## 4.2.9.3.4.6.7 OData Request: Delete Entity

Create an OData request to delete an entity in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

The **DeleteEntity Request** enables an **EntityType** instance to be deleted from a data service. The base rules and semantics of this request type are defined by AtomPub, as specified in [RFC5023](#) section 9.4 -- Deleting Resources with **DELETE**.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

To delete an individual entity, the client makes a **DELETE** request to a URL that identifies the entity.

When the delete is successfully completed, the response MUST be **<response code and name>** and contain an empty body.

## Example Requests

### Version 4

Delete the employee with id '**007**' of entity set "**Employees**"

```
DELETE /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
```

### Version 2

Delete the employee with id '**007**' of entity set "**Employees**"

```
DELETE /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees('007')
```

# Delete Entity Request

## Overview

### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **DELETE** entity request.

The starting point for a **DELETE** entity request is the Client Proxy instance. You can create an entity resource based on an entity list resource, which you can use to create a delete request.

## Example

Delete the employee with key "id = '007'" of the Version 4 entity set '**Employees**':

```
DELETE /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('007')
with IF-MATCH header W/"MI6".
```

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_delete_request  TYPE REF TO /iwbep/if_cp_request_delete,
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.
lo_entity_resource
= lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES')-
>navigate_with_key(value /iwbep/if_v4_tea_busi_types=>ty_s_employee_key( id =
'007' ) .
lo_delete_request = lo_entity_resource->create_request_for_delete( ).
lo_delete_request->set_if_match( 'mi6' ).
lo_delete_request->execute( ).
lo_delete_request->check_execution( ).
```

## Steps

**Step 1:** Create the entity resource for the employee with the key "Id='007'" of entity set '**Employees**' (with internal name '**EMPLOYEES**'). Type "**ty\_s\_employee\_key**" in interface "**/iwbep/if\_v4\_tea\_busi\_types**" is a structure containing the key property "id":

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.
lo_entity_resource
= lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES')-
>navigate_with_key(value /iwbep/if_v4_tea_busi_types=>ty_s_employee_key( id =
'007' ) .
```

**Step 2:** Create the delete request instance on the entity resource:

```
DATA: lo_delete_request  TYPE REF TO /iwbep/if_cp_request_delete,
      lo_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.
lo_delete_request = lo_entity_resource->create_request_for_delete( ).
```

**Step 3:** Set the If-Match header to provide an ETag (if needed for this request). If the corresponding request header is, for example, **if-match: W/"MI6"**, the required input for **SET\_IF\_MATCH** is '**MI6'**:

```
DATA: lo_delete_request TYPE REF TO /iwbep/if_cp_request_delete.
lo_delete_request->set_if_match( 'mi6' ).
```

**Step 4:** Run the delete request:

```
DATA: lo_delete_request TYPE REF TO /iwbepl/if_cp_request_delete.  
lo_delete_request->execute( ).
```

#### ⓘ Note

A successful delete request doesn't have response data, which means that the **EXECUTE** method doesn't return a delete response object.

**Step 5:** Check if the execution was successful:

```
DATA: lo_delete_request TYPE REF TO /iwbepl/if_cp_request_delete.  
lo_delete_request->check_execution( ).
```

#### ⓘ Note

Normally, any issues during the request execution raises an exception in method **EXECUTE**. This method call is optional.

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.8 OData Request: Read Entity List

Create an OData request to read an entity list (entity collection) in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A **RetrieveEntitySet** Request is used by a client to update the entries in an AtomPub Collection, as specified in [RFC5023](#), that maps to an **EntitySet** in the abstract data model.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

OData services support data requests for using HTTP GET requests. The URL path specifies the target of the request (for example, the collection of entities, entity, navigation property, structural property, or operation).

## Example Requests

### Version 4

Get all entities of entity set “**Employees**”

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees
```

### Version 2

Get all entities of entity set “**Employees**”

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees
```

## Read Entity List Request

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to create a **READ** request on an entity list.

The starting point for a **GET** entity request is the [Client Proxy Examples \[page 918\]](#). You can create a **read list request** based on an **entity list resource**.

Running the **read list request** provides the read list response that can provide the business data of the READ entity list request.

### Example

Fetch all entities of the Version 4 entity set ‘**Employees**’:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees
```

```
DATA: lo_client_proxy          TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_read_list_request    TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_read_list_response   TYPE REF TO /iwbep/if_cp_response_read_lst,
      lo_entity_list_resource TYPE REF TO /iwbep/if_cp_resource_list,
      lt_employee             TYPE STANDARD TABLE OF /iwbep/s_v4_tea_employee.

      lo_entity_list_resource = lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES' ).
      lo_read_list_request = lo_entity_list_resource->create_request_for_read( ).
      lo_read_list_response = lo_read_list_request->execute( ).
      lo_read_list_response->get_business_data( IMPORTING et_business_data =
      lt_employee ).
```

### Steps

**Step 1:** Create the entity list resource for entity set ‘**Employees**’ (with **internal** name ‘**EMPLOYEES**’):

```
DATA: lo_client_proxy          TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_entity_list_resource TYPE REF TO /iwbep/if_cp_resource_list.
```

```
lo_entity_list_resource = lo_client_proxy->create_resource_for_entity_set( 'EMPLOYEES' ).
```

**Step 2:** Create the read list request instance on the entity resource:

```
DATA: lo_read_list_request      TYPE REF TO /iwbepl/if_cp_request_read_list,
      lo_entity_list_resource TYPE REF TO /iwbepl/if_cp_resource_list.
      lo_read_list_request = lo_entity_list_resource->create_request_for_read( ).
```

**Step 3:** Run the read list request and get the read list response instance:

```
DATA: lo_read_list_request      TYPE REF TO /iwbepl/if_cp_request_read_list,
      lo_read_list_response TYPE REF TO /iwbepl/if_cp_response_read_lst.
      lo_read_list_response = lo_read_list_request->execute( ).
```

**Step 4: Fetch the business data from the response object:**

```
DATA: lo_read_list_response TYPE REF TO /iwbepl/if_cp_response_read_lst,
      lt_employee           TYPE STANDARD TABLE OF /iwbepl/s_v4_tea_employee.
      lo_read_list_response->get_business_data( IMPORTING et_business_data =
      lt_employee ).
```

#### ⓘ Note

When you're using the Version 4 local client proxy, the business data you receive from the local consumption response isn't converted again for outbound processing.

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.9 OData Request: Update Entity List

Create an OData request to update an entity list in the Client Proxy instance.

You want

## OData Specification

### OData Version 2

An update request for an entity list is not supported in OData Version 2.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

You can update entity collections by submitting a **PATCH** request to the collection's resource path. The body of the request:

- **MUST** be a **delta payload**.
- the resource path of the collection **MUST NOT** include type cast or filter segments.
- **MUST NOT** include any system query options that affect the shape of the result.

Added/changed entities are applied as upserts. Deleted entities are applied as deletions.

## Example Requests

### Version 4

Update the complete entity set “**Teams**”:

```
PATCH /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams
```

With request body

```
{
  "@context" : "#$delta",
  "value" : [
    {
      "Id" : "NEW_1",
      "Name" : "Created",
      "MemberCount" : 2,
      "ManagerId" : "3",
      "BudgetCurrency" : "USD",
      "Budget" : 555.55
    },
    {
      "Id" : "TEAM_02",
      "Name" : "Update?"
    },
    {
      "Id" : "New_3",
      "Name" : "Created"
    }
  ]
}
```

## Updata Entity List Request

### Overview

The starting point for a **PATCH** entity request is the Client Proxy instance. You can create an **entity resource** based on an **update list** request, and use it to create an update request.

Running the **update list** provides the update list response, which can return the business data of the **UPDATE** entity list request.

## Example

Update the entity set “**Teams**”:

```
PATCH /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams
```

With request body

```
{  
    "@context" : "#$delta",  
    "value" : [  
        {  
            "Id" : "New Team",  
            "Name" : "Created",  
            "BudgetCurrency" : "USD",  
            "Budget" : 555  
        },  
        {  
            "Id" : "TEAM_02",  
            "Name" : "New Team Name"  
        }  
    ]  
}
```

```
DATA: lo_client_proxy TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_update_list_request TYPE REF TO /iwbep/if_cp_request_update_1,  
      lo_update_list_response TYPE REF TO /iwbep/if_cp_response_update_1,  
      lo_list_resource TYPE REF TO /iwbep/if_cp_resource_list,  
      lt_response_data TYPE STANDARD TABLE OF /iwbep/s_v4_tea_team,  
      lt_request_data TYPE STANDARD TABLE OF /iwbep/s_v4_tea_team.  
      lo_list_resource = lo_client_proxy->create_resource_for_entity_set( 'TEAMS' ).  
      lo_update_list_request = lo_list_resource->create_request_for_update( ).  
      lt_request_data = value #( ( id ='New team' name ='Created' budget =555  
      budget_currency ='USD' )  
          ( id = 'TEAM_02' name = 'New Team Name' ) ).  
      lo_update_list_request->set_business_data( it_business_data =  
      lt_request_data ).  
      lo_update_list_request->request_continue_on_error( ).  
      lo_update_list_response = lo_update_list_request->execute( ).  
      lo_update_list_response->get_business_data( IMPORTING et_business_data  
      = lt_response_data )
```

## Steps

**Step 1:** Create the entity list resource for entity set ‘**Teams**’ (with **internal** name ‘**TEAMS**’):

```
DATA: lo_client_proxy TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_list_resource TYPE REF TO /iwbep/if_cp_resource_list.  
      lo_list_resource = lo_client_proxy->create_resource_for_entity_set( 'TEAMS' ).
```

**Step 2:** Create the update list request instance on the entity list resource:

```
DATA: lo_update_list_request TYPE REF TO /iwbep/if_cp_request_update_1,  
      lo_list_resource TYPE REF TO /iwbep/if_cp_resource_list.  
      lo_update_list_request = lo_list_resource->create_request_for_update( ).
```

**Step 3:** Set the business data (for example, update) for your update list request. Define the corresponding business data and set it into the update list request:

```
DATA: lo_update_list_request TYPE REF TO /iwbep/if_cp_request_update_1,  
      lt_request_data TYPE STANDARD TABLE OF /iwbep/s_v4_tea_team.
```

```

lt_request_data = VALUE #( ( id = 'New team' name = 'Created' budget =
555 budget_currency = 'usd' )
( id = 'team_02' name = 'New Team Name' ) ) .
lo_update_list_request->set_business_data( it_business_data =
lt_request_data ).
```

**Step 4:** (Optional) Request that the request processing continues even when there is an error (if supported).

If the server supports this setting, a failed operation for one entity doesn't cause the complete request to fail. The failed entity is marked with the instance annotation `Core.DataModificationException`:

```
DATA:     lo_update_list_request TYPE REF TO /iwbp/if_cp_request_update_l.
        lo_update_list_request->request_continue_on_error( ).
```

### ⓘ Note

This information is mapped into the request header `prefer:odata.continue-on-error`. This preference is optional. The consumed OData service does **not** have to follow this preference.

**Step 5:** Run the update list request and get the update list response instance:

```
DATA:     lo_update_list_request  TYPE REF TO /iwbp/if_cp_request_update_l,
        lo_update_list_response TYPE REF TO /iwbp/if_cp_response_update_l.
        lo_update_list_response = lo_update_list_request->execute( ).
```

**Step 6:** Fetch the business data from the response list object:

```
DATA:     lo_update_list_response TYPE REF TO /iwbp/if_cp_response_update_l,
        lt_response_data          TYPE STANDARD TABLE OF /iwbp/s_v4_tea_team.
        lo_update_list_response->get_business_data( IMPORTING et_business_data
= lt_response_data ).
```

## Constraints

- Updating an entity list is only supported for OData Version 4
- This feature is not supported for local consumption
- Only **PATCH** is supported. **PUT** is not supported.
- **IF-MATCH** handling is not supported
- Query options are not supported
- Updating an entity list within a \$batch is not supported

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.10 OData Request: Custom Query Option

Create an OData request using a custom query option that isn't one of the OData-defined system query options in the Client Proxy instance.

### OData Specification

#### OData Version 2

See also: [\[MS-O DATA\]: Open Data Protocol \(OData\)](#).

Custom Query Options enable you to include data service-specific information in a data service URI query string.

#### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

Services can support additional custom query options that are not defined in the OData specification. The custom query options:

- can't begin with the "\$" or "@" character
- can't conflict with any OData-defined system query options defined in the OData version that the service supports

### Example Requests

#### Version 4

Use custom query option “**sap-statistics**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams?sap-statistics=true
```

#### Version 2

Use custom query option “**sap-statistics**”:

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Prices?sap-statistics=true
```

# Custom Query Option Request

## Overview

### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use `$count`.

The starting point for a custom query option request is an entity list read request. You can set a custom query option on the entity list read request.

## Example

Use custom query option “`sap-statistics`”

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams?sap-statistics=true

DATA: lo_read_list_request      TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_http_client          TYPE REF TO if_http_client,
      lv_sap_statistics_result TYPE string.
      lo_read_list_request->set_custom_query_options( VALUE #( ( name = 'sap-statistics' value = 'true' ) ) ).
      lo_read_list_request->execute( ).
      lv_sap_statistics_result = lo_http_client->response->get_header_field( name = 'sap-statistics' ).
```

## Steps

**Step 1:** Set the custom query option directly at the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.
      lo_read_list_request->set_custom_query_options( VALUE #( ( name = 'sap-statistics' value = 'true' ) ) ).
```

**Step 2:** Run the read request and fetch the sap statistics result from the HTTP client instance:

```
DATA: lo_read_list_request      TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_http_client          TYPE REF TO if_http_client,
      lv_sap_statistics_result TYPE string.
      lo_read_list_request->execute( ).
      lv_sap_statistics_result = lo_http_client->response->get_header_field( name = 'sap-statistics' ).
```

### ⓘ Note

`lo_http_client` must be the **same** HTTP client instance as the one used for creating the Client Proxy.

## Constraints

- Regardless of the Client Proxy OData version, input MUST conform with the more stricter OData Version 4 specification.

- You can only use custom query options for remote consumption. Custom query options are not supported in local consumption scenarios.

## Related Information

[OData Request Terms \[page 930\]](#)

[Client Proxy Instance Types \[page 904\]](#)

## 4.2.9.3.4.6.11 OData Request: Deep Create

Create an OData request to execute a “deep create” (deep insert) in the Client Proxy instance.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

In the Deep Create request, you can insert a new EntityType instance (E1) into an EntitySet and insert new entities related to E1. This is described by a NavigationProperty on the EntityType associated with E1) in a single InsertEntity Request. For example, in a customer relationship management-focused data service, you can insert a new customer entity and new related order entities in a single **InsertEntity** Request. This type of an InsertEntity Request is also known as a "deep insert".

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

A "deep insert" request creates an entity that includes related entities using the appropriate inline representation

## Example Requests

### Version 4

Create a new team (entity set “**TEAMS**”) and a new manager (entity set “**MANAGERS**”) via navigation property “**TEAM\_2\_MANAGER**”:

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/TEAMS
```

With request body

```
{
    "Name" : "Business Suite",
```

```

    "Member_Count" : 2,
    "Manager_ID" : "8",
    "BudgetCurrency" : "USD",
    "Budget" : 555.55,
    "Team_2_Manager" : {
        "ID" : "8",
        "Team_ID" : ""
    }
}

```

## Version 2

Create the employee with id '1' (entity set “**EMPLOYEES**”) and the corresponding team (entity set “**TEAMS**”) with navigation property “**My\_Team**”:

```
POST /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees
```

With request body

```
{
    "Location" : {
        "Country" : "Germany",
        "City" : {
            "PostalCode" : "69124",
            "CityName" : "Heidelberg",
            "VeryLongPropertyNameForOrderBy" : ""
        }
    },
    "Id" : "1",
    "Name" : "Walter Winter",
    "Age" : "52",
    "EntryDate" : "\/Date(915148800000)\/",
    "Manager_ID" : "1",
    "Room_ID" : "1",
    "Team_ID" : "TEAM_01",
    "My_Team" : {
        "Team_Identifier" : "TEAM_01",
        "Name" : "Business Suite"
    }
}
```

## Deep Create Request

### Overview

#### Note

As the coding itself is independent of the OData version, we will just present one general example on how to create a execute a deep create request.

As the coding is independent of the OData version, we're presenting a general example on how to create a **deep create** request on an entity.

The main difference between a **create** and a **deep create** request in the Client Proxyinstance is that the **deep create** request requires a data description node. You create a data description node at the create request instance and it is used later when setting the deep business data.

## ⓘ Note

See [OData Request: Create Entity \[page 939\]](#) for more information.

## Example

Create a new team (entity set “**Teams**”) and a new manager (entity set “**Managers**”):

```
POST /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Team
```

With request body

```
{  
    "Id" : "TEAM_04",  
    "MemberCount" : 2,  
    "Team2Manager" : {  
        "Id" : "0009"  
    }  
}
```

```
DATA: lo_create_request      TYPE REF TO /iwbep/if_cp_request_create,  
      lo_create_response     TYPE REF TO /iwbep/if_cp_response_create,  
      lo_data_des_node_root  TYPE REF TO /iwbep/if_cp_data_desc_node,  
      lo_data_desc_node_child TYPE REF TO /iwbep/if_cp_data_desc_node,  
      ls_deep_busi_data      TYPE /iwbep/  
if_v4_tea_busi_types=>ty_s_team_and_manager,  
      ls_response_data       TYPE /iwbep/  
if_v4_tea_busi_types=>ty_s_team_and_manager.  
      lo_data_desc_node_root = lo_create_request->  
>create_data_description_node( ).  
      lo_data_desc_node_root->set_properties( VALUE #( ( | ID| ) ( |  
MEMBER_COUNT| ) ) ).  
      lo_data_desc_node_child = lo_data_desc_node_root-  
>add_child( 'team_2_manager' ).  
      lo_data_desc_node_child->set_properties( VALUE #( ( | ID| ) ) ).  
      ls_deep_busi_data = VALUE #( id = 'team_04'  
      member_count = 2  
      team_2_manager = VALUE #( id = '0009' ) ).  
      lo_create_request->set_deep_business_data( is_business_data =  
ls_deep_busi_data  
      io_data_description_node = lo_data_desc_node_root ).  
      lo_create_response = lo_create_request->execute( ).  
      lo_create_response->get_business_data( IMPORTING es_business_data =  
ls_response_data ).  
      lo_create_response->get_business_data( IMPORTING es_business_data =  
ls_response_data ).
```

## Steps

**Step 1:** Creation of the data description node for the root entity (here: “Teams”) on the create request instance:

```
DATA: lo_create_request      TYPE REF TO /iwbep/if_cp_request_create,  
      lo_data_des_node_root  TYPE REF TO /iwbep/if_cp_data_desc_node.  
      lo_data_desc_node_root = lo_create_request->  
>create_data_description_node( ).
```

## ⓘ Note

See [OData Request: Create Entity \[page 939\]](#) for more information.

**Step 2:** Set the properties for the root entity. In our example, the properties are “**Id**” (internal name “**ID**”) and “**MemberCount**” (internal name “**MEMBER\_COUNT**”):

```
DATA:    lo_data_des_node_root TYPE REF TO /iwbepl/if_cp_data_desc_node.
        lo_data_desc_node_root->set_properties( VALUE #( ( |ID| ) ( | MEMBER_COUNT| ) ) ).
```

**Step 3:** Create the data description node for the child entity (in this example, “**Managers**”) on the root data description node. “**TEAM\_2\_MANAGER**” is the **internal** name of navigation property “**Team2Manager**”:

```
DATA:    lo_data_des_node_root    TYPE REF TO /iwbepl/if_cp_data_desc_node,
        lo_data_desc_node_child TYPE REF TO /iwbepl/if_cp_data_desc_node.
        lo_data_desc_node_child = lo_data_desc_node_root-
>add_child( 'team_2_manager' ).
```

**Step 4:** Set the properties for the child entity (in this example, “**Managers**”). Only property “**Id**” (with internal name “**ID**”) is included:

```
DATA:    lo_data_desc_node_child TYPE REF TO /iwbepl/if_cp_data_desc_node.
        lo_data_desc_node_child->set_properties( VALUE #( ( |ID| ) ) ).
```

**Step 5:** Define the request data for the deep create and set the deep business data into the request instance (in the root data description node):

```
DATA:    lo_create_request      TYPE REF TO /iwbepl/if_cp_request_create,
        lo_data_des_node_root TYPE REF TO /iwbepl/if_cp_data_desc_node,
        ls_deep_busi_data     TYPE /iwbepl/
if_v4_tea_busi_types=>ty_s_team_and_manager.
        ls_deep_busi_data = VALUE #( id = 'team_04'
            member_count = 2
            team_2_manager = VALUE #( id = '0009' ) ).
        lo_create_request->set_deep_business_data( is_business_data =
ls_deep_busi_data
        io_data_description_node = lo_data_desc_node_root ).
```

### ⓘ Note

The property that references the child entity business data MUST be named using the **internal** name of the corresponding navigation property (in this example, the property is “**TEAM\_2\_MANAGER**”, which describes the business data for the child entity “**Managers**”).

### ⓘ Note

If the underlying entity has value control or VCS properties. The VCS properties **should** also be part of the provided data container (in this example, **ls\_deep\_busi\_data**). If the properties aren't part of the data container, the behavior (for example conversion exits) is undefined and can cause unexpected errors.

**Step 6:** Run the deep create request and fetch the response business data from the create response instance:

```
DATA:    lo_create_request    TYPE REF TO /iwbepl/if_cp_request_create,
        lo_create_response   TYPE REF TO /iwbepl/if_cp_response_create,
        ls_response_data    TYPE /iwbepl/
if_v4_tea_busi_types=>ty_s_team_and_manager.
        lo_create_response = lo_create_request->execute( ).
        lo_create_response->get_business_data( IMPORTING es_business_data =
ls_response_data ).
```

### Note

This part is identical to handling a typical create request in the Client Proxy instance.

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.12 OData Request: Delta Link Query Option

Create an OData request with \$delta token query option in the Client Proxy instance.

## OData Specification

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

Delta links are service-generated links the client uses to access newly created, updated, or deleted entities without a full read of the target resource with every request.

Delta links are based on a defining query that tracks the changes of the set of results. For example, the request that generated the results containing the delta link. The delta link encodes the entity collection that tracks the changes. Also, it includes a starting point to begin track changes.

## Example Requests

### Version 4

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_tech/0001/PagedDeltaEntities?  
$deltatoken=20170612000000
```

## Delta Link Query Request

### Overview

The starting point for a delta link in the Client Proxy instance is either a Client Proxy instance or a read list request instance (depending on your use case).

## Example 1: Create a new delta link

You ran a read list request in the Client Proxy instance and you want to save a delta link for this response, so you can track future changes to the current response:

**Step 1:** Save the current response as delta link (in this example under the name '**DELTA\_LINK\_NAME**'):

```
DATA: lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.  
lo_read_list_response->save_delta_link( 'delta_link_name' ).
```

## Example 2: Create a delta request from an existing delta link

You have an existing delta link and want to use it to create a delta request:

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_delta_read_request TYPE REF TO /iwbep/if_cp_request_read_dlta,  
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.  
CHECK lo_client_proxy->does_delta_link_exist( 'delta_link_name' ) = abap_true.  
CHECK lo_client_proxy->can_delta_request_be_created( 'delta_link_name' ) =  
abap_true.  
lo_delta_read_request = lo_client_proxy->create_request_for_delta( 'delta_link_name' ).  
lo_read_list_response = lo_delta_read_request->execute( ).
```

## Steps

**Step 1:** Check that the delta link:

- does exist (in this example, '**DELTA\_LINK\_NAME**').
- can be used to create a delta request. If the stored delta link was manually changed, the delta link can't be used).

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_delta_read_request TYPE REF TO /iwbep/if_cp_request_read_dlta,  
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.  
CHECK lo_client_proxy->does_delta_link_exist( 'delta_link_name' ) = abap_true.  
CHECK lo_client_proxy->can_delta_request_be_created( 'delta_link_name' ) =  
abap_true.
```

### ⓘ Note

You can only find and use delta links that you created.

**Step 2:** Use the stored delta link to create a delta link request instance:

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,  
      lo_delta_read_request TYPE REF TO /iwbep/if_cp_request_read_dlta.  
lo_delta_read_request = lo_client_proxy->create_request_for_delta( 'delta_link_name' ).
```

**Step 3:** Run the delta link request and fetch the corresponding response instance:

```
DATA: lo_delta_read_request TYPE REF TO /iwbep/if_cp_request_read_dlta,  
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.  
lo_read_list_response = lo_delta_read_request->execute( ).
```

### Example 3: Insert an existing delta link into an existing request

You have an existing request and want to use it in combination with an existing delta link:

```
DATA: lo_client_proxy      TYPE REF TO /iwbep/if_cp_client_proxy,
      lo_read_list_request  TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.
CHECK lo_client_proxy->does_delta_link_exist('delta_link_name') = abap_true.
CHECK lo_client_proxy->can_delta_request_be_created('delta_link_name') =
abap_false.
lo_read_list_request->use_delta_link('delta_link_name').
lo_read_list_response = lo_read_list_request->execute().
```

### Steps

**Step 1:** Check that the delta link:

- does exist (in this example, '**DELTA\_LINK\_NAME**').
- can be used to create a delta request. If it can be used to create a delta request, you use method **CREATE\_REQUEST\_FOR\_DELTA**, in example 2.

```
DATA: lo_client_proxy TYPE REF TO /iwbep/if_cp_client_proxy.
CHECK lo_client_proxy->does_delta_link_exist('DELTA_LINK_NAME') = abap_true.
CHECK lo_client_proxy->can_delta_request_be_created('DELTA_LINK_NAME') =
abap_false.
```

#### ⓘ Note

You can only find and use delta links that you created.

**Step 2:** Insert the delta link into the existing read list request:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.
lo_read_list_request->use_delta_link('delta_link_name').
```

**Step 3:** Get the read list response instance using the read list request

```
DATA: lo_read_list_request  TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst.
lo_read_list_response = lo_read_list_request->execute().
```

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Read Entity List \[page 953\]](#)

[Client Proxy Instance Types \[page 904\]](#)

## 4.2.9.3.4.6.13 OData Request: Using Navigation

Create an OData request using a navigation in the Client Proxy instance.

### OData Specification

#### OData Version 2

See also: [\[MS-O DATA\]: Open Data Protocol \(OData\)](#).

A unidirectional (one-way) relationship (for example, a Link) is when two entity types are related by association, but only one of the entity types defines a **NavigationProperty** that binds to the association.

#### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

Relationships from one entity to another are represented as navigation properties. Navigation properties are defined as part of an entity type, but can also appear on entity instances as undeclared dynamic navigation properties. Each relationship has a cardinality.

### Example Requests

#### Version 4

Get all employees associated with Team '**TEAM\_01**' via navigation property '**Team2Employees**':

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams('TEAM_01')/Team2Employees
```

#### Version 2

Get the team of the employee with Id '**0005**' via navigation property "**My\_Team**":

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees(Id='0005')/My_Team
```

## Using Navigation Request

### Overview

#### ① Note

As the coding is independent of the OData version, we're presenting a general example on how to use navigations.

The starting point for a request using a navigation is an entity resource. On the entity resource, you can create a resource for the corresponding navigation target.

## Example 1

Navigate to an entity list with Navigation Property “**Department2Teams**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/
Departments(Id='1',Sector='Consulting')/Department2Teams
```

### Steps

**Step 1:** Create the target resource on the entity resource using the **internal** name of the navigation property (“**DEPARTMENT\_2\_TEAMS**”):

```
DATA: lo_entity_resource      TYPE REF TO /iwbep/if_cp_resource_entity,
      lo_target_list_resource TYPE REF TO /iwbep/if_cp_resource_list.
      lo_target_list_resource = lo_entity_resource->
navigate_to_many( 'department_2_teams' ).
```

#### ⓘ Note

You can use the target resource to create a READ request.

## Example 2

Navigate to a single entity using the Navigation Property “**Team2Manager**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams('TEAM_01')/
Team2Manager
```

### Steps

**Step 1:** Create the target resource on the entity resource using the **internal** name of the navigation property (“**TEAM\_2\_MANAGER**”):

```
DATA: lo_entity_resource      TYPE REF TO /iwbep/if_cp_resource_entity,
      lo_target_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity.
      lo_target_entity_resource = lo_entity_resource->
navigate_to_single( 'team_2_manager' ).
```

#### ⓘ Note

You can use the target resource to create a READ request.

## Example 3

Navigate to an optional (for example, zero-to-one navigation) entity using the Navigation Property “**Employee2Manager**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0004')/
Employee2Manager
```

### Step-by-step

**Step 1:** Create the target resource on the entity resource using the **internal** name of the navigation property (“**EMPLOYEE\_2\_MANAGER**”):

```
DATA: lo_entity_resource      TYPE REF TO /iwbep/if_cp_resource_entity,
      lo_target_entity_resource TYPE REF TO /iwbep/if_cp_resource_entity01.
```

```
lo_target_entity_resource = lo_entity_resource->  
navigate_to_optional( 'employee_2_manager' ).
```

### ⓘ Note

You can use the target resource to create an optional READ request.

## Related Information

[OData Request: Read Entity \[page 946\]](#)

## 4.2.9.3.4.6.14 OData Request: Including a Nextlink

Create an OData request to read an entity list (entity collection) with a next link in the Client Proxy instance.

## OData Specification

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

In responses that include a partial set of the items identified by the request, the URL MUST include a link (a next link) that allows retrieving the next partial set of items. A next link representation is format-specific. The final partial set of items MUST NOT contain a next link.

## Nextlink Request

### Overview

The starting point for a **next link** request is the Client Proxy instance, is **read list** response instance.

### Example

You created a read list request in the Client Prox instance and want to manage potential next links:

```
DATA: lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,  
      lt_employee          TYPE STANDARD TABLE OF /iwbep/s_v4_tea_employee.  
      lo_read_list_response->get_business_data( IMPORTING et_business_data =  
      lt_employee ).  
      WHILE lo_read_list_response->has_next( ).  
        lo_read_response = lo_read_list_response->get_next( ).  
        lo_read_list_response->get_business_data( IMPORTING et_business_data =  
        lt_employee ).  
      ENDWHILE.
```

## Steps

**Step 1:** Fetch the first batch of entities from the read list response instance:

```
DATA: lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,  
      lt_employee          TYPE STANDARD TABLE OF /iwbep/s_v4_tea_employee.  
      lo_read_list_response->get_business_data( IMPORTING et_business_data =  
      lt_employee ).
```

**Step 2:** Check that there are still next links using the **HAS\_NEXT** method on the response instance). Then you can fetch the next batch of entities. Get the corresponding response instance by using the **GET\_NEXT** method on the previous response instance. Then get the business data from this new instance:

```
DATA: lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,  
      lt_employee          TYPE STANDARD TABLE OF /iwbep/s_v4_tea_employee.  
      WHILE lo_read_list_response->has_next( ).  
        lo_read_response = lo_read_response->get_next( ).  
        lo_read_list_response->get_business_data( IMPORTING et_business_data =  
      lt_employee ).  
      ENDWHILE.
```

## Constraints

- Next links are only supported for OData Version 4 requests.
- Next links are only supported for remote consumption.

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Read Entity List \[page 953\]](#)

## 4.2.9.3.4.6.15 OData Request System Query Options

Learn how to use OData system query options. System query options include:

### 4.2.9.3.4.6.15.1 \$expand Option

Use the `$expand` system query option to represent associated EntityType instance or EntitySet inline.

## OData Specification

### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

The `$expand` System Query Option indicates that entities associated with the EntityType instance or EntitySet (identified by the Resource Path section of the URI) must be represented inline.

### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

The `$expand` system query option indicates the related entities and stream values that MUST be represented inline. The service MUST return the specified content and can choose to return additional information.

The value of the `$expand` query option is a comma-separated list of navigation property names, stream property names, or `$value` that indicate the stream content of a media-entity.

## Example Requests

### Version 4

Get the team '`TEAM_03`' and its manager with navigation property "`Team2Manager`":

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams('TEAM_03')?  
$expand=Team2Manager
```

### Version 2

Get the manager with id '`0004`' and the information about his team with navigation property "`My_Team`":

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Managers('0001')?$expand=My_Team
```

# \$expand System Query Option

## Overview

### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$expand** option.

The starting point for a request with the **\$expand** option is an entity list read request. On the entity list read request, you can set the **\$expand** option.

## Example

Get all entities from the entity set “**Teams**”. For each term, fetch the corresponding manager using the navigation property “**Team2Manager**”. You want to retrieve only the Id of each manager:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams?
$expand=Team2Manager($select=Id)
```

## Steps

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_expand_node_root  TYPE REF TO /iwbep/if_cp_expand_node,
      lo_expand_node       TYPE REF TO /iwbep/if_cp_expand_node.
lo_expand_node_root = lo_read_list_request->create_expand_node( ).
lo_expand_node = lo_expand_node_root->add_expand( 'team_2_manager' ).
lo_expand_node->set_select_properties( VALUE #( ( CONV #( 'id' ) ) ) ).
```

### ⓘ Note

The **SET\_EXPAND** method is obsolete. It's not necessary to explicitly set the expand node in the request.

**Step 1:** Create the root expand node on the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_expand_node_root  TYPE REF TO /iwbep/if_cp_expand_node,
      lo_expand_node_root = lo_read_list_request->create_expand_node( ).
```

**Step 2:** Add the expand path by using the **internal** name of the navigation property (“**TEAM\_2\_MANAGER**”). This step creates a new expand node:

```
DATA: lo_expand_node_root TYPE REF TO /iwbep/if_cp_expand_node,
      lo_expand_node       TYPE REF TO /iwbep/if_cp_expand_node,
      lo_expand_node = lo_expand_node_root->add_expand( 'team_2_manager' ).
```

**Step 3:** Set the selected properties (using **internal** names) on the select node. “**ID**” is the internal name of property “**Id**”:

```
DATA: lo_expand_node TYPE REF TO /iwbep/if_cp_expand_node,
      lo_expand_node->set_select_properties( VALUE #( ( CONV #( 'id' ) ) ) ).
```

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Read Entity \[page 946\]](#)

[OData Request: Read Entity List \[page 953\]](#)

## 4.2.9.3.4.6.15.2 \$filter Option

Use the \$filter system query option to restrict the returned set of items.

## OData Specification

### OData Version 2

See also: [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A data service URI with a **\$filter** System Query Option identifies a subset of the entities in the EntitySet (identified by the Resource Path section of the URI) by only selecting the entities that meet the predicate expression the query option specifies.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol](#).

The \$filter system query option restricts the set of items that are returned.

## Example Requests

### Version 4

Get the entities in entity set “**Employees**” that have the Id “**0006**” and the Postalcode “**69190**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees?  
$filter=Location/City/Postalcode eq '69190' and Id eq '0006'
```

### Version 2

Get the entities in entity set “**Employees**” with the Id “**0006**” and the PostalCode “**69190**”:

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees?$filter=Location/City/  
PostalCode eq '69190' and Id eq '0006'
```

# \$filter System Query Option

## Overview

### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$filter** option.

The starting point for a request with the **\$filter** option is an entity list read request.

## Example

Get all entities from the entity set “**Buildings**” with the BuildingID “**ROT05**” and Cityname is not “**Walldorf**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Buildings?
$filter=Location/City/Cityname ne 'Walldorf' and BuildingID eq 'ROT05'
```

## Steps

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,
      lt_range           TYPE RANGE OF string,
      lo_filter_factory  TYPE REF TO /iwbep/if_cp_filter_factory,
      lo_filter_node_1    TYPE REF TO /iwbep/if_cp_filter_node,
      lo_filter_node_2    TYPE REF TO /iwbep/if_cp_filter_node,
      lo_filter_node_final TYPE REF TO /iwbep/if_cp_filter_node.
      lo_filter_factory = lo_read_list_request->create_filter_factory( ).
      lt_range = VALUE #( ( option = 'ne' sign = 'i' low = 'Walldorf' ) ).
      lo_filter_node_1 = lo_filter_factory->create_by_range( iv_property_path =
      'location-city-cityname'
                                         it_range = lt_range ).
      lt_range = VALUE #( ( option = 'eq' sign = 'i' low = 'rot05' ) .
      lo_filter_node_2 = lo_filter_factory->create_by_range( iv_property_path =
      'building_id'
                                         it_range = lt_range ).
      lo_filter_node_final = lo_filter_node_1->and( lo_filter_node_2 ).
      lo_read_list_request->set_filter( lo_filter_node_final ).
```

**Step 1:** Create an instance of the filter factory at the read list request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_filter_factory      TYPE REF TO /iwbep/if_cp_filter_factory.
      lo_filter_factory = lo_read_list_request->create_filter_factory( ).
```

**Step 2:** Create a filter node for the first filter expression (`Location/City/Cityname ne 'Walldorf'`). We define the new values and set the business data to the request:

Properties	Internal Names
Cityname	CITYNAME
Location	LOCATION
City	CITY

The filter expression must be expressed in a range:

```
DATA: lt_range           TYPE RANGE OF string,
      lo_filter_factory  TYPE REF TO /iwbep/if_cp_filter_factory,
```

```

lo_filter_node_1  TYPE REF TO /iwbep/if_cp_filter_node.
lo_filter_factory = lo_read_list_request->create_filter_factory( ).
lt_range = VALUE #( ( option = 'ne' sign = 'i' low = 'Walldorf' ) ).
lo_filter_node_1 = lo_filter_factory->create_by_range( iv_property_path =
'location-city-cityname'
                                         it_range =
lt_range
).

```

**Step 3:** Create a filter node for the second filter expression **BuildingID' ROT05'**. The internal name of the primitive property “**BuildingID**” is “**BUILDING\_ID**”. The filter expression must be expressed in a range:

```

DATA: lt_range          TYPE RANGE OF string,
      lo_filter_factory TYPE REF TO /iwbep/if_cp_filter_factory,
      lo_filter_node_2  TYPE REF TO /iwbep/if_cp_filter_node.
      lt_range = VALUE #( ( option = 'eq' sign = 'i' low = 'rot05' ) ).
      lo_filter_node_2 = lo_filter_factory->create_by_range( iv_property_path =
'building_id'
                                         it_range =
lt_range
).

```

**Step 4:** Connect the two filter nodes with “**and**” in the final filter node:

```

DATA: lo_filter_node_1      TYPE REF TO /iwbep/if_cp_filter_node,
      lo_filter_node_2      TYPE REF TO /iwbep/if_cp_filter_node,
      lo_filter_node_final  TYPE REF TO /iwbep/if_cp_filter_node.
      lo_filter_node_final = lo_filter_node_1->and( lo_filter_node_2 ).

```

**Step 5:** Set the final **filter** node in the request instance:

```

DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,
      lo_filter_node_final TYPE REF TO /iwbep/if_cp_filter_node.
      lo_read_list_request->set_filter( lo_filter_node_final ).

```

## Negation

To use negation call the **NOT** method on a **filter** node instance (for example, **\$filter=not(BuildingID is 'WDF03')**). This step creates a new filter node (the negated filter node):

```

DATA: lo_filter_node      TYPE REF TO /iwbep/if_cp_filter_node,
      lo_filter_node_not TYPE REF TO /iwbep/if_cp_filter_node.
      lo_filter_node_not = lo_filter_node->not( ).

```

## Constraints

- **\$filter** is only supported for primitive and complex properties.
- Only these range options are allowed: **EQ**, **NE**, **GT**, **GE**, **LT**, and **LE**.
- Only range signs “**I**” and “**E**” are allowed.
- Conversions with range option “**CP**” are not allowed.
- Only these functions can be used with **CP: startswith**, **endswith**, and **substringof** (for OData Version 2), **contains** (for OData Version 4).
- Not all filter expression are supported for local consumption.
- The currency code must be provided when the addressed property has a reference to a currency property.
- The unit of measurement must be provided when the addressed property has a reference to a unit property

- You can't set a **\$filter** for an **\$expand**.

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/TEAMS?
$expand=TEAM_2_EMPLOYEES($filter=AGE eq 56)
```

## Related Information

[OData Request Terms \[page 930\]](#)

### 4.2.9.3.4.6.15.3 \$count Option

Use the **\$count** or **\$inlinecount** system query option to indicate a certain number or total count of entities in the EntitySet.

## OData Specification

### OData Version 2

See also: [\[MS-O DATA\]: Open Data Protocol \(OData\) ↗](#).

**\$inlinecount**: For a value of "**allpages**", the **\$inlinecount** option indicates the response to the request must include the number of entities count in the EntitySet. The number of entities count is identified by the Resource Path section of the URI after all **\$filter** System Query Options are applied.

If the value is "**none**", this option indicates the response to the request MUST NOT include the count value.

### OData Version 4

See also: [OData Version 4.01. Part 1: Protocol ↗](#).

The **\$count** system query option with a value of **true** specifies that the total item count in a collection that matches the request is returned with the result.

The **\$count** system query option ignores **\$top**, **\$skip**, or **\$expand** query options, and returns the total results count across all pages (including only the results that match any specified **\$filter** and **\$search**). The count returned inline might not equal the actual number of items returned. This happens due to latency between calculating the count and enumerating the last value or due to inexact calculations on the service.

## Example Requests

### Version 4

Get the number of entities in entity set "**TEAMS**":

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/TEAMS/$count
```

Get all Equipment entities associated with "**Employee\_1**" and the total count of associated items:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES('1')/EMPLOYEE_2_EQUIPMENTS?$count=true
```

## Version 2

Get the number of entities in entity set "**EquipmentSet**":

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/EquipmentSet/$count
```

Get all Team Member entities associated with Team "**TEAM\_01**" and the total count of associated members:

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Teams('TEAM_01')/Team_Members?$inlinecount=allpages
```

# \$count System Query Option

## Overview

### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$count** option.

The starting point for a request with the **\$count** option is an entity list read request. You can set the **\$count** on the entity list read request.

You can fetch the **\$count** result from the entity list read response instance.

### ⓘ Note

The actual "flavor" of the **\$count** that the Client Proxy sets depends on the request type:

- **\$count** if no business data is requested.
- **\$count=true** for OData Version 4 requests if business data is requested.
- **\$inlinecount=allpages** for OData Version 2 if business data is requested.

## Example 1

Get the number of entities in entity set "**TEAMS**":

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/TEAMS/$count
```

## Steps

```
DATA: lo_read_list_request  TYPE REF TO /iwbep/if_cp_request_read_list,
       lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,
       lv_count               TYPE REF TO int8.
lo_read_list_request->request_no_business_data( ).
lo_read_list_request->request_count( ).
lo_read_list_response = lo_read_list_request->execute( ).
lv_count = lo_read_list_response->get_count( ).
```

**Step 1:** Set the request to NOT fetch the entity list (for example, `request_no_business_data`):

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
lo_read_list_request->request_no_business_data( ).
```

**Step 2:** Request the `$count` at the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
lo_read_list_request->request_count( ).
```

**Step 3:** Run the request and fetch the count value from the read list response instance:

```
DATA: lo_read_list_request  TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,  
      lv_count              TYPE REF TO int8.  
lo_read_list_response = lo_read_list_request->execute( ).  
lv_count = lo_read_list_response->get_count( ).
```

## Example 2

Get all `Equipment` entities associated with "`Employee 1`" and the total count of associated items:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES('1')/  
EMPLOYEE_2_EQUIPMENTS?$count=true
```

## Steps

```
DATA: lo_read_list_request  TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_read_list_response TYPE REF TO /iwbep/if_cp_response_read_lst,  
      lv_count              TYPE REF TO int8.  
lo_read_list_request->request_count( ).  
lo_read_list_response = lo_read_list_request->execute( ).  
lv_count = lo_read_list_response->get_count( ).
```

### ⓘ Note

The steps for this example are identical to the first example, except that you do **NOT** request to not return any business data (you skip step 1 in the first example).

See example one for the steps.

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Read Entity \[page 946\]](#)

[OData Request: Read Entity List \[page 953\]](#)

## 4.2.9.3.4.6.15.4 \$orderby Option

Use the \$orderby system query option to determine what values are used to order the entities in the EntitySet.

### OData Specification

#### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A data service URI with the **\$orderby** System Query Option specifies an expression for determining what values are used to order the entities in the EntitySet (identified by the Resource Path section of the URI).

#### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

The **\$orderby** System Query option specifies the order that items are returned from the service. The value of the **\$orderby** System Query option is a comma-separated list of expressions that use the primitive result values to sort the items. The expression can include the suffix **asc** for ascending or **desc** for descending that you separate from the property name by one or more spaces.

### Example Requests

#### Version 4

Get all entities of entity set “**Employees**” and sort the response descending to property “**Cityname**” and ascending to property “**Age**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees?  
$orderby=Location/City/Cityname desc, Age
```

#### Version 2

Get all entities of entity set “**Employees**” and sort the response ascending to property “**Id**” and descending to property “**Name**”:

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Employees?$orderby=Id, Name asc
```

## \$orderby System Query Option

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$orderby** option.

The starting point for a request with the **\$orderby** option is an entity list read request. You can set the **\$orderby** on the entity list read request.

## Example

Get all entities of entity set “**Employees**” and sort the response descending to property “**Cityname**” (in complex property “**City**”, which is part of complex property “**Location**”) and ascending to property “**Age**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees?  
$orderby=Location/City/Cityname desc, Age
```

## Steps

**Step 1:** Set the **\$orderby** values at the request instance:

Properties	Internal Names
Age	<b>AGE</b>
Location	<b>LOCATION</b>
City	<b>CITY</b>
Cityname	<b>CITYNAME</b>

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
      lo_read_list_request-> set orderby( value #( ( property_path = 'LOCATION-  
CITY-CITYNAME'  
          descending = abap_true )  
        ( property_path =  
          'AGE'  
          descending =  
          abap_false ) ) ).
```

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Read Entity \[page 946\]](#)

[OData Request: Read Entity List \[page 953\]](#)

## 4.2.9.3.4.6.15.5 \$select Option

Use the \$select system query option to return a subset for the returned properties of the URI without a \$select query option.

### OData Specification

#### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A data service URI **with** a **\$select** System Query Option identifies the same set of entities as a URI **without** a **\$select** query option. If you have a **\$select** query option, the data service response returns a subset (identified by the **\$select** query option) for the returned properties of the URI that didn't include a **\$select** query option.

#### OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

The **\$select** system query option requests that the service return only the properties, dynamic properties, actions, and functions explicitly requested by the client. The service returns the specified content (if available) and any available expanded navigation or stream properties. It can also return additional information.

### Example Requests

#### Version 4

Get the employee with Id '**0002**' of entity set "**Employees**" and return only properties "**Name**" and "**Age**":

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0002')?  
$select=Name,Age
```

#### Version 2

Get all entities of entity set "**Managers**" and return only properties "**Name**" and "**Age**":

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Managers?$select=Name,Age
```

## \$select System Query Option

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$select** option.

The starting point for a request with **\$select** is a read request on an entity list. You can set the selected properties on the entity list read request.

## Example

Set the properties “**Age**”, “**Name**” and “**Cityname**” for the employee with Id ‘**0002**’ for the entity set “**Employees**”. “**Cityname**” is in the complex property “**City**”, which is of the complex property “**Location**”:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Employees('0002')?  
$select=Name, Age, Location/City/Cityname
```

## Steps

**Step 1:** Set the properties you want to select at the read request instance:

Properties	Internal Names
Age	AGE
Location	LOCATION
City	CITY
Cityname	CITYNAME
Name	NAME

```
DATA: lo_read_request TYPE REF TO /iwbep/if_cp_request_read.  
      lo_read_request->set_select_properties( VALUE #( ( CONV #('AGE') )  
                                              ( CONV #('LOCATION-CITY-CITYNAME') )  
                                              ( CONV #('NAME') ) ) ).
```

## Related Information

[OData Request Terms \[page 930\]](#)

[OData Request: Create Entity \[page 939\]](#)

[OData Request: Read Entity \[page 946\]](#)

## 4.2.9.3.4.6.15.6 \$skip and \$top Options

Use the \$skip or \$top system query options to identify a subset of the entities in an entity collection.

## OData Specification

### OData Version 2

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

A data service URI with:

<b>\$skip System Query Option</b>	<b>\$top System Query Option</b>
identifies a subset of the entities in an entity collection (identified by the Resource Path section of the URI)	identifies a subset of the entities in an entity collection (identified by the Resource Path section of the URI)
the subset is defined by searching <b>N</b> entities in an collection and selecting <b>only the remaining entities</b> (starting with entity <b>N+1</b> ).	the subset is formed by selecting <b>only the first N</b> items of the set.
<b>N</b> is a positive integer specified by this query option.	<b>N</b> is a positive integer specified by this query option.

## OData Version 4

See [OData Version 4.01. Part 1: Protocol](#).

A data service URI with:

<b>\$skip System Query Option</b>	<b>\$top System Query Option</b>
specifies a non-negative integer <b>n</b> that <b>excludes</b> the first <b>n</b> items of the queried collection.	specifies a non-negative integer <b>n</b> that <b>limits</b> the number of items returned from a collection.
the service returns items starting at position <b>n+1</b> .	the service returns the number of available items up to but not greater than the specified value <b>n</b> .

## Example Requests

### Version 4

Skip the first entity in entity set “**TEAMS**” and get the following two ones:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams?$skip=1&$top=2
```

### Version 2

Skip the first entity in entity set “Managers” and get the following two ones:

```
GET /sap/opu/odata/IWBEP/TEA_TEST_APPLICATION/Managers?$skip=1&$top=2
```

## \$skip and \$top Query Options

### Overview

#### ⓘ Note

As the coding is independent of the OData version, we're presenting a general example on how to use the **\$count** option.

The starting point for a request with **\$skip** or **\$top** option is an entity list read request. You can set both the **\$skip** and the **\$top** on the entity list read request.

## Example

Skip the first entity in entity set “**TEAMS**” and get these entities:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0003/Teams?$skip=1&$top=2
```

## Steps

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
      lo_read_list_request->set_skip( 1 ).  
      lo_read_list_request->set_top( 2 ).
```

**Step 1:** Set the **\$skip** value at the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
      lo_read_list_request->set_skip( 1 ).
```

**Step 2:** Set the **\$top** value at the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
      lo_read_list_request->set_top( 2 ).
```

## Related Information

[OData Request Terms \[page 930\]](#)

## 4.2.9.3.4.6.15.7 \$search Option

Use the **\$search** system query option to restrict results to include items you specify in the expression..

## OData Specification

### OData Version 4

See [\[MS-ODATA\]: Open Data Protocol \(OData\)](#).

The **\$search** system query option restricts the result to include only the items that match the specified search expression. The type of match depends on your implementation.

## Example Requests

### Version 4

Get all entities of Entity Set “**Employees**” with “**Peter**” or “**Smith**” in their name:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES?$search=Peter  
OR Smith
```

## \$search System Query Option

### Overview

The starting point for a request with the **\$search** option is an entity list read request. You can set the **\$search** expression on the entity list read request.

#### ⓘ Note

Depending on your implementation, the result of the **\$search** option on a property can vary. In our example, **\$search** acts on property “**Name**”.

### Example

Search all entities of Entity Set “**Employees**” with “**Peter**” or “**Smith**” in their name:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES?  
$search=Peter OR Smith
```

### Steps: Option 1

**Step 1:** Set the search expression at the request instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list.  
lo_read_list_request->set_search( 'Peter OR Smith' ).
```

#### ⓘ Note

This approach is only supported for remote consumption, not for local consumption

### Steps: Option 2

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_search_node      TYPE REF TO /iwbep/if_cp_search_node.  
lo_search_node = lo_read_list_request->create_search_node( 'Peter' ).  
lo_search_node->or( 'Smith' ).  
lo_read_list_request->set_search_node( lo_search_node ).
```

**Step 1:** Create the search node on the request instance. For this step, you already provided the first part of your **\$search** expression (“**Peter**” from “**Peter OR Smith**”):

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_search_node      TYPE REF TO /iwbep/if_cp_search_node.
```

```
lo_search_node = lo_read_list_request->create_search_node( 'Peter' ).
```

**Step 2:** Insert the second expression ("Smith") and connect it with "OR" to the first expression:

```
DATA: lo_search_node TYPE REF TO /iwbep/if_cp_search_node.  
lo_search_node->or( 'Smith' ).
```

**Step 3:** Set the search node in the request:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_search_node      TYPE REF TO /iwbep/if_cp_search_node.  
      lo_read_list_request->set_search_node( lo_search_node ).
```

## Negation

Negate the previous expression. For example, this request:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES?$search=NOT  
(Peter OR Smith)
```

Call the method "NOT" on the search node instance:

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_search_node      TYPE REF TO /iwbep/if_cp_search_node.  
      lo_search_node = lo_read_list_request->create_search_node( 'Peter' ).  
      lo_search_node->or( 'Smith' ).  
      lo_search_node->not( ).  
      lo_read_list_request->set_search_node( lo_search_node ).
```

## Connect Two Search Nodes

You can connect two search nodes. For example, this request:

```
GET /sap/opu/odata4/iwbep/tea/default/iwbep/tea_busi/0001/EMPLOYEES?  
$search=(Peter OR Smith) AND (John AND Mary)
```

```
DATA: lo_read_list_request TYPE REF TO /iwbep/if_cp_request_read_list,  
      lo_search_node_1      TYPE REF TO /iwbep/if_cp_search_node,  
      lo_search_node_2      TYPE REF TO /iwbep/if_cp_search_node.  
      lo_search_node_1 = lo_read_list_request->create_search_node( 'Peter' ).  
      lo_search_node_1->or( 'Smith' ).  
      lo_search_node_2 = lo_read_list_request->create_search_node( 'John' ).  
      lo_search_node_2->and( 'Mary' ).  
      lo_search_node_1->and_node( lo_search_node_2 ).  
      lo_read_list_request->set_search_node( lo_search_node_1 ).
```

## Constraints

- **\$search** is only available for OData Version 4 requests.
- **SET\_SEARCH** is not supported for local consumption.

## Related Information

[OData Request Terms \[page 930\]](#)

### 4.2.9.3.5 HTTP Outbound Communication

Learn how to set up and enable HTTP outbound communication.

HTTP outbound connectivity enables you to consume HTTP services from other systems. To enable HTTP outbound communication, there are two different stages that are performed by different roles:

#### Development Tasks

Development tasks are performed by developers. Development tasks are all tasks that are performed within ABAP development tools for Eclipse, such as:

- Creating a Communication Scenario
- Creating an outbound service

For more information, see [Enable HTTP Communication in Your ABAP Code \[page 989\]](#).

#### Administrator Tasks

Administrator tasks are performed by administrators. Administrator tasks are all tasks that are performed in SAP Fiori launchpad, such as:

- Creating a communication system
- Creating a communication arrangement
- Creating communication users

For more information, see [Set Up HTTP Communication \[page 1003\]](#).

[Enable HTTP Communication in Your ABAP Code \[page 989\]](#)

The HTTP client allows to connect to HTTP endpoints.

[Set Up HTTP Communication \[page 1003\]](#)

To set up HTTP communication, use the corresponding communication management apps.

## 4.2.9.3.5.1 Enable HTTP Communication in Your ABAP Code

The HTTP client allows to connect to HTTP endpoints.

### Feature Scope

To enable HTTP communication, there are four different approaches:

- Communication target
  - Provides APIs for receiver validation based on arbitrary properties (see [Define Specific Properties for Communication Scenarios \[page 894\]](#)).
  - Local configuration via communication management UIs
  - Requires the development of communication scenario artifacts.
- Communication arrangement
  - Provides APIs for receiver validation based on arbitrary properties (see [Define Specific Properties for Communication Scenarios \[page 894\]](#)).
  - Local configuration via communication management UIs or central configuration in SAP BTP cockpit via SAP BTP destination service reference in the communication system (not relevant for Developer Extensibility, see [Developer Extensibility](#)). See [Create HTTP Destinations](#) and [Communication Systems \[page 2598\]](#).
  - Requires the development of communication scenario artifacts.
- Destination service (not relevant for Developer Extensibility, see [Developer Extensibility](#)).
  - Provides APIs to fetch connection information for a given destination name from the SAP BTP destination service.
  - No additional development artifacts required.
  - Central configuration in SAP BTP cockpit.
  - Comparable to `create_by_destination` in on-premise ABAP.
- Static configuration in code (only recommended for public services and test purposes).

We recommend using the communication target approach for the following reasons:

- There are no hard-coded values/assumptions for a destination name.
- You have more control over which applications can use a particular application destination.
- The application coding is independent of the deployment (Cloud or On-Premise).
- It allows to define application specific destinations.
- It ensures content separation between software components.

#### Note

When using on-premise connectivity, make sure the called ICF service is exposed in Cloud Connector.

#### [HTTP Communication via Communication Targets \[page 990\]](#)

HTTP communication can be established using so-called communication targets.

#### [HTTP Communication via Communication Arrangements \[page 994\]](#)

HTTP outbound communication can be established using so-called communication arrangements.

[HTTP Communication via Destination Service \(Deprecated\) \[page 996\]](#)

[HTTP Communication via URL \[page 999\]](#)

[Example: HTTP Multipart Request \[page 999\]](#)

[Example: Enable Path Prefix \[page 1001\]](#)

[Example: Retry Design Pattern \[page 1002\]](#)

## 4.2.9.3.5.1.1 HTTP Communication via Communication Targets

HTTP communication can be established using so-called communication targets.

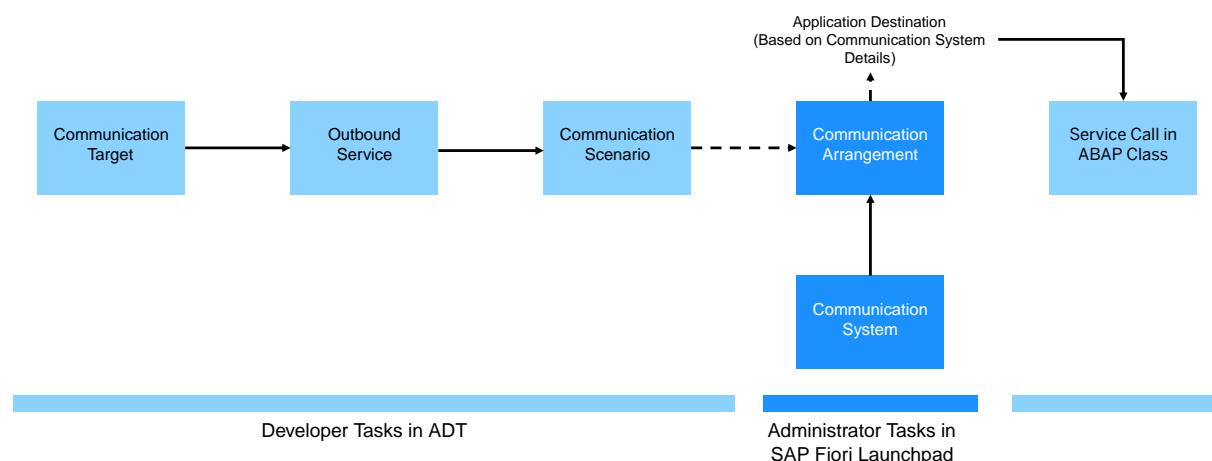
[Using Communication Targets \[page 990\]](#)

[Calling an HTTP Service via Communication Targets \[page 991\]](#)

### 4.2.9.3.5.1.1.1 Using Communication Targets

For outbound HTTP communication of newly created applications, we recommend using communication targets and application destinations. With this approach, developers can control which applications can use a particular application destination. This process is called receiver validation.

The creation of an application destination is triggered when a communication arrangement is created based on a communication scenario that references an outbound service of type communication target.



## Communication Targets With Exactly One Assigned Application Destination

The communication target can have either multiple or only exactly one assigned application destination. To call an HTTP service via communication targets in your ABAP code, create an HTTP client from your communication target class and call its execute method.

## Communication Target With Several Assigned Application Destinations

If the communication target has several assigned application destinations, additionally, the following is required:

- When creating the communication target instance, pass the name of the relevant application destination to the parameter `application_destination`.
- Before, you need to get the name of the relevant application destination. We recommend storing the application destinations assigned to a communication target in a customizing table. To do so, create an implementation for the BAdI `communication_management` as described in [Calling an HTTP Service via Communication Targets \[page 991\]](#). From the customizing table, select the relevant application destination in your service call.

### ⓘ Note

- An application destination is always uniquely assigned to a communication target. When the communication target is created, a unique global communication target class with the same name is generated.
- An application can only use an application destination if it has permission to access the corresponding communication target class. This communication class is assigned to the application destination. This is only possible if this application is in the same package, or if the communication target class is part of the relevant package interface or main package interfaces. The violation of a main package or software component interface leads to a runtime error, while other package interface violations lead to warnings.

### 4.2.9.3.5.1.1.2 Calling an HTTP Service via Communication Targets

#### Prerequisites

You've created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).

## Context

To call an HTTP service via communication targets, proceed as follows.

## Procedure

1. Create a communication target as described in [Working With Communication Targets](#).
2. Create a corresponding outbound service of type communication target as described in [Creating Outbound Services](#).
3. Add the relevant communication target to the newly created outbound service.
4. Add the newly created outbound service to your scenario.
5. Publish the communication scenario. The administrator can then create the required communication management objects as described in [Next Steps \[page 994\]](#).
6. If the communication target has multiple assigned application destinations, create an implementation of the communication\_management BAdl as described in the example. The BAdl implementation stores the name of the application destinations in a e.
7. Adapt the service call according to your needs as described in the example.

## Example

Create a Customizing table to store the name of the application destination.

### Sample Code

```
@EndUserText.label : 'Cota Custom Table'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #C  
@AbapCatalog.dataMaintenance : #RESTRICTED  
define table zcota_custom {  
    key client : abap.clnt not null;  
    key usecase : abap.intl not null;  
    appldest : abap.sstring(50) not null;  
}
```

Retrieve the name of the relevant application destination.

### Sample Code

```
METHOD if_com_arrangement_badi~during_save.  
*  
*      !io_com_arrangement TYPE REF TO if_com_arrangement  
*      !io_com_system      TYPE REF TO if_com_system  
*      !io_com_user        TYPE REF TO if_com_user  
*      !io_com_scenario     TYPE REF TO if_com_scenario  
DATA ls_custom TYPE zcota_custom.  
DATA(lt_out_srv) = io_com_scenario->get_outbound_services( ).  
DATA(lt_out_srv_a) = io_com_arrangement->get_outbound_services( ).
```

```

FIELD-SYMBOLS: <ls_out_srv> LIKE LINE OF lt_out_srv_a.
DATA(lt_properties) = io_com_arrangement->get_properties( ).
DATA lv_scenarioid TYPE i VALUE 1.
LOOP AT lt_properties INTO DATA(ls_property).
  IF ls_property-name EQ 'USECASE'.
    lv_scenarioid = ls_property-values[ 1 ].
  ENDIF.
ENDLOOP.
" save the application destination to an application-specific persistence
LOOP AT lt_out_srv_a ASSIGNING <ls_out_srv>.
  ls_custom-usecase = lv_scenarioid.
  ls_custom-appldest = <ls_out_srv>-destination_name.
  MODIFY zcota_custom FROM @ls_custom.
ENDLOOP.
ENDMETHOD.

METHOD if_com_arrangement_badi~before_delete.
*   ! @parameter io_com_arrangement | <p class="shorttext synchronized"
lang="en">Communication arrangement reference</p>
*   ! @parameter io_com_system | <p class="shorttext synchronized"
lang="en">Communication system reference</p>
*   ! @parameter io_com_user | <p class="shorttext synchronized"
lang="en">Communication user reference</p>
*   ! @parameter io_com_scenario | <p class="shorttext synchronized"
lang="en">Communication scenario reference</p>
*   ! @parameter et_message | <p class="shorttext synchronized"
lang="en">Messages</p>

DATA(lt_out_srv) = io_com_arrangement->get_outbound_services( ).

FIELD-SYMBOLS: <ls_out_srv> LIKE LINE OF lt_out_srv.

LOOP AT lt_out_srv ASSIGNING <ls_out_srv>.
  DELETE FROM zcota_custom WHERE appldest = @<ls_out_srv>-
destination_name.
ENDLOOP.

ENDMETHOD.

```

## HTTP service call

### ↔ Sample Code

```

METHOD if_oo_adt_classrun~main.
  DATA lo_cota TYPE REF TO z_cota_http.
  DATA lt_output TYPE STANDARD TABLE OF string.
  DATA lv_appl_dest TYPE sappdestname.
  DATA(lv_scen) = '1'.
  " select the relevant application destination from the Customizing table
  SELECT SINGLE appldest
  FROM zcota_custom
  WHERE usecase = @lv_scen
  INTO @lv_appl_dest.
  TRY.
    lo_cota = NEW z_cota_http( lv_appl_dest ).
    DATA(lo_client) = lo_cota->create_web_http_client( ).
    DATA(lo_response) = lo_client->execute( if_web_http_client=>get ).
    DATA(ls_status) = lo_response->get_status( ).
    CATCH cx_appdestination INTO DATA(lx_appdestination).
      APPEND lx_appdestination->get_text( ) TO lt_output.
    CATCH cx_communication_target_error INTO
DATA(lx_communication_target_error).
      APPEND lx_communication_target_error->get_text( ) TO lt_output.
    CATCH cx_web_http_client_error INTO DATA(lx_web_http_client_error).
      APPEND lx_web_http_client_error->get_text( ) TO lt_output.
  ENDTRY.
  out->write( lo_response->get_text( ) ).

```

```
ENDMETHOD.
```

## Next Steps

To test your outbound call, you have to provide a configuration for the outbound service you want to call in your code.

Create a communication system and communication arrangement for the communication scenario and maintain the required data, such as host name and credentials in the Communication Systems and Communication Arrangements apps. For more information, see [Communication Management \[page 883\]](#).

These tasks are performed by the administrator. For more information, see [Set Up HTTP Communication \[page 1003\]](#).

### 4.2.9.3.5.1.2 HTTP Communication via Communication Arrangements

HTTP outbound communication can be established using so-called communication arrangements.

#### Prerequisites

You've created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).

#### Procedure

1. Create a corresponding outbound service of type HTTP.
  1. In your ABAP project, select the relevant package node in the Project Explorer.
  2. Open the context menu and choose      .
  3. Enter the name of the outbound service.
  4. Select *HTTP Service* in the *Service Type* dropdown list.
  5. Choose *Next* and select a transport request.
  6. Optional: Go to your outbound service and enter the URL in the field *Default Path Prefix*.
  7. Save the outbound service.
2. Add the newly created outbound service to your scenario.
3. Publish the communication scenario. The administrator can then create the required communication management objects as described in [Test Your Outbound Call \[page 996\]](#).

- Adapt the communication arrangement call according to your needs as described in the example. See [Service Consumption via Communication Arrangements \[page 892\]](#) for more information. According to your developed communication scenario, add the following:

comm_scenario	mandatory	ID of the developed communication scenario.
comm_system_id	optional	ID of the configured communication system. Use method <code>query_ca</code> of class <code>cl_com_arrangement_factory</code> to derive it dynamically.
service_id	optional	ID of the developed outbound service.

### ⓘ Note

You can't use the `create_by_comm_arrangement` method for SAP-delivered scenarios.

## Example

### ↴ Sample Code

```

DATA: lr_cscn type if_com_scenario_factory=>ty_query-cscn_id_range.

" find CA by scenario
  lr_cscn = value #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ) .
  DATA(lo_factory) = cl_com_arrangement_factory=>create_instance( ).
  lo_factory->query_ca(
    EXPORTING
      is_query           = value #( cscn_id_range = lr_cscn )
    IMPORTING
      et_com_arrangement = data(lt_ca) ).

  IF lt_ca is initial.
    EXIT.
  ENDIF.

" take the first one
  READ TABLE lt_ca INTO DATA(lo_ca) INDEX 1.

" get destination based to Communication Arrangement
  TRY.
    DATA(lo_dest) =
    cl_http_destination_provider=>create_by_comm_arrangement(
      comm_scenario  = '<Scenario ID>'
      service_id     = '<Outbound Service ID>'
      comm_system_id = lo_ca->get_comm_system_id( ) ).

    DATA(lo_http_client) =
    cl_web_http_client_manager=>create_by_http_destination( lo_dest ).

    " execute the request
      DATA(lo_request) = lo_http_client->get_http_request( ).
      DATA(lo_response) = lo_http_client-
    >execute( if_web_http_client=>get ).

    CATCH cx_http_dest_provider_error.
  
```

```
    " handle exception here
    CATCH cx_web_http_client_error.
        " handle exception here
    ENDTRY.
```

### ⓘ Note

We recommend to retrieve the correct destination reference based on the communication scenario and a customer-defined property as described in [Service Consumption via Communication Arrangements \[page 892\]](#).

## Test Your Outbound Call

To test your outbound call, you have to provide a configuration for the outbound service you want to call in your code.

Create a communication system and communication arrangement for the communication scenario and maintain the required data, such as host name and credentials in the Communication Systems and Communication Arrangements apps. For more information, see [Communication Management \[page 883\]](#).

These tasks are performed by the administrator. For more information, see [Set Up HTTP Communication \[page 1003\]](#).

## Related Information

[Set Up HTTP Communication \[page 1003\]](#)

### 4.2.9.3.5.1.3 HTTP Communication via Destination Service (Deprecated)

## Context

Use the destination service in SAP BTP to store destination information that can be reused by applications deployed in one of the BTP environments.

### ⓘ Note

You can use the destination service. However, this approach is deprecated. We recommend using the communication arrangement approach instead. See [HTTP Communication via Communication Arrangements \[page 994\]](#) for more information.

You've the following options to consume a destination service:

- Dynamic by using a communication arrangement and a communication system
- Static by using the method `create_by_cloud_destination` in the code

## Procedure

When using `create_by_cloud_destination`, proceed as follows:

Create a destination object using class `cl_http_destination_provider` and method `create_by_cloud_destination` with the following parameters:

- `i_name`: the name of the destination
- Optional: `i_service_instance_name`: Typically, you use the destinations of the subaccount in which the ABAP instance resides or, in case of a SaaS solution based on the ABAP environment, the destinations of the consumer subaccount. However, you can add more destinations using your own destination service instance and communication scenario `SAP_COM_0276`, for example, to achieve separation of concerns (see also [Create a Destination \[page 891\]](#)). In this case, specify the value of the service instance name property of the communication arrangement for `SAP_COM_0276`.
- `i_authn_mode`: Set the value of this parameter according to the authentication method configured in your destination. If the authentication method uses user propagation, the value is `if_a4c_cp_service=>user_propagation`, if it doesn't, set `if_a4c_cp_service=>service_specific`:

Authentication Methods

Value	Proxy Type <i>Internet</i>	Proxy Type <i>OnPremise</i>
<code>if_a4c_cp_service=&gt;user_propagation</code>	<ul style="list-style-type: none"><li>• OAuth User Token Exchange Authentication</li><li>• OAuth SAML Bearer Assertion Authentication</li><li>• OAuth JWT Bearer Authentication</li></ul>	Principal Propagation SSO Authentication for HTTP
<code>if_a4c_cp_service=&gt;service_specific</code>	<ul style="list-style-type: none"><li>• No Authentication</li><li>• Basic Authentication</li><li>• <a href="#">Client Authentication Types for HTTP Destinations</a></li><li>• OAuth Client Credentials Authentication</li><li>• <a href="#">OAuth Password Authentication</a></li></ul>	<ul style="list-style-type: none"><li>• No Authentication</li><li>• Basic Authentication</li></ul>

Other authentication methods are currently not supported by the SAP BTP, ABAP environment.

## Example

The actual processing of an HTTP request and its response is shown in the following code example:

### ↳ Sample Code

```
DATA lo_http_destination TYPE REF TO if_http_destination.
DATA lo_http_client      TYPE REF TO if_web_http_client.
DATA lo_http_response    TYPE REF TO if_web_http_response.

TRY.
  " create HTTP destination by cloud destination
  lo_http_destination =
cl_http_destination_provider=>create_by_cloud_destination( i_name =
'DestinationName' i_authn_mode = if_a4c_cp_service=>service_specific ).

  " create HTTP client by HTTP destination
  lo_http_client =
cl_web_http_client_manager=>create_by_http_destination( lo_http_destination ).

  " adding header fields
  lo_http_client->get_http_request( )->set_header_fields( VALUE
#( ( name = if_web_http_header=>content_type value =
if_web_http_header=>accept_application_json )

( name = if_web_http_header=>accept           value =
if_web_http_header=>accept_application_json ) ) .

  " execute HTTP GET-request and store response
  lo_http_response = lo_http_client->execute( if_web_http_client=>get ).

  " print response text to console
  DATA(ls_status) = lo_http_response->get_status( ).
  out->write( |Response is: { ls_status-code } { ls_status-reason }.| ).
  out->write( lo_http_response->get_text( ) ).

  CATCH cx_http_dest_provider_error cx_web_http_client_error INTO
DATA(lx_error).
  " display error details
  out->write( lx_error->get_text( ) ).

ENDTRY.
```

## Test Your Outbound Call

To test your outbound call, configure an HTTP destination as described in [Create HTTP Destinations](#).

### Authentication Methods

- Only the authentication methods listed in table *Authentication Methods* are available.
- If you're using Client Certificate Authentication as authentication method, activate [\*Use client provided certificate\*](#). This flag is only visible if the URL field contains a URL string starting with <https://...>.
- For proxy type **Internet** and the use of authentication type **client certificate**, you must upload the X.509 client certificate in P12 format on the client side in the SAP BTP, ABAP environment using the Maintain Client Certificates application. For more information, see [Maintain Client Certificates \[page 2772\]](#). Uploading the client certificate via destination service isn't supported.

- The use of `if_a4c_cp_service=>user_propagation` isn't supported in the ADT class runner. It can only be tested when a business user context is available, for example, during processing of OData services or HTTP services.

## Related Information

[Destination Service \[page 885\]](#)

### 4.2.9.3.5.1.4 HTTP Communication via URL

You can configure the URL static in the code:

#### ↔ Sample Code

```
DATA(lo_url_destination) = cl_http_destination_provider->create_by_url(
    'https://
<host>/sap/bc/srt/xip/sap/<provider>/000/<service>/<binding>').
```

#### ⓘ Note

Using `create_by_url` is only suitable for public services or test purposes, because credentials should be stored in the communication management in SAP Fiori launchpad.

## Related Information

[Set Up HTTP Communication \[page 1003\]](#)

### 4.2.9.3.5.1.5 Example: HTTP Multipart Request

Using multipart requests for HTTP communication from the ABAP environment.

The code examples below show an **HTTP multipart request**:

## Client Side

#### ↔ Sample Code

```
DATA: http_client TYPE REF TO if_web_http_client,
```

```

lo_response TYPE REF TO if_web_http_response,
iv_url TYPE string.

iv_url = 'https://....'. "Enter a correct url

TRY.
  http_client =
cl_web_http_client_manager=>create_by_http_destination( i_destination =
cl_http_destination_provider=>create_by_url( i_url = iv_url ) ).

DATA(lo_request) = http_client->get_http_request(   ).
lo_request->set_header_field( i_name = 'Content-type'
                                i_value = 'multipart/mixed' ).

DATA(part_1) = lo_request->add_multipart(   ).
part_1->set_header_field( i_name = 'Content-type'
                           i_value = 'text/html; charset=UTF-8' ).

part_1->set_text( 'This is part one.' ).

DATA(part_2) = lo_request->add_multipart(   ).
part_2->set_header_field( i_name = 'Content-type'
                           i_value = 'text/html; charset=UTF-8' ).
part_2->set_text( 'This is part two.' ).

lo_response = http_client->execute( if_web_http_client=>post ).

DATA(status) = lo_response->get_status(   .
IF status-code NE 200.
  "Error handling here
ENDIF.

CATCH cx_web_http_client_error cx_http_dest_provider_error.
  "Handle exception here.
ENDTRY.

```

## Server Side

### ↔ Sample Code

```

CLASS ZCL_TEST_MULTIPART IMPLEMENTATION.
method IF_HTTP_SERVICE_EXTENSION~HANDLE_REQUEST.
  DATA: answer TYPE string,
        num_part TYPE i.

  num_part = request->num_multiparts(   .
  IF num_part = 0.
    answer = '<html><body>No multipart found in request!!</body></html>'.
    response->set_text(  answer ).
  ELSE.
    DO num_part TIMES.
      DATA(lo_part_request) = request->get_multipart( index = sy-index ).
      IF lo_part_request IS BOUND.
        "Do something here with this part.
      ENDIF.
    ENDDO.
  ENDIF.
ENDMETHOD.
ENDCLASS.

```

## 4.2.9.3.5.1.6 Example: Enable Path Prefix

Enable a path prefix for HTTP calls in ABAP.

Interface `IF_WEB_HTTP_CLIENT` provides a method `enable_path_prefix` to extend a URL set by the method `set_uri_path`.

Calling only `set_uri_path` overwrites the URL used to instanciate the client:

### ↳ Sample Code

```
DATA: http_client TYPE REF TO if_web_http_client,
      lo_response TYPE REF TO if_web_http_response,
      iv_url TYPE string.

      iv_url = 'https://hostfoobar:port/fool/bar1'. "Enter a correct url

      TRY.
         http_client =
         cl_web_http_client_manager->create_by_http_destination( i_destination =
         cl_http_destination_provider->create_by_url( i_url = iv_url ) ).

         DATA(lo_request) = http_client->get_http_request(   ).

         lo_request->set_uri_path( EXPORTING i_uri_path = '/foo2/bar2' ). "the request will be send to /foo2/bar2
         lo_response = http_client->execute( if_web_http_client=>post ).

         DATA(status) = lo_response->get_status( ).
         IF status-code NE 200.
            "Error handling here
         ENDIF.

         CATCH cx_web_http_client_error cx_http_dest_provider_error.
            " handle exception here
         ENDTRY.
```

If you call the method `enable_path_prefix` before calling `set_uri_path`, the path passed by `set_uri_path` is appended to the initial path:

### ↳ Sample Code

```
DATA: http_client TYPE REF TO if_web_http_client,
      lo_response TYPE REF TO if_web_http_response,
      iv_url TYPE string.

      iv_url = 'https://hostfoobar:port/fool/bar1'. "Enter a correct url

      TRY.
         http_client =
         cl_web_http_client_manager->create_by_http_destination( i_destination =
         cl_http_destination_provider->create_by_url( i_url = iv_url ) ).

         DATA(lo_request) = http_client->get_http_request(   ).

         http_client->enable_path_prefix( ).
```

```

lo_request->set_uri_path( EXPORTING i_uri_path = '/foo2/bar2' ).
"the request will be send to /foo1/bar1/foo2/bar2
lo_response = http_client->execute( if_web_http_client=>post ).

DATA(status) = lo_response->get_status( ).
IF status-code NE 200.
  "Error handling here
ENDIF.

CATCH cx_web_http_client_error cx_http_dest_provider_error.
  " handle exception here
ENDTRY.

```

#### 4.2.9.3.5.1.7 Example: Retry Design Pattern

Repeat an HTTP request automatically in ABAP.

Using this feature, you can repeat a request automatically up to 3 times, as long as the status code from the response has an appropriate value.

You can provide a set of response codes upon which the request is supposed to be repeated, if the corresponding response returns a status code from this set.

If you do not provide this set of codes, a default set is drawn depending on the idempotence of a request. You can control the idempotence of a request via parameter.

The default set for a non-idempotent request is 408, 429 and 503. For an idempotent request, the default set is 308, 429, 500, 502, 503, 504, 507, 509.

All other parameters correspond to the known `execute` method.

##### Sample Code

```

DATA: http_client          TYPE REF TO if_web_http_client,
      lo_response        TYPE REF TO if_web_http_response,
      lt_retry_status_codes  TYPE http_status_codes,
      iv_url             TYPE string.
iv_url = 'https://hostfoobar:port/foo/bar'. "Enter a correct url
TRY.
  http_client =
    cl_web_http_client_manager=>create_by_http_destination( i_destination =
    cl_http_destination_provider=>create_by_url( i_url = iv_url ) ).

  "select for which numbers the execution should be retried
  lt_retry_status_codes = VALUE http_status_codes( ( '111' )
                                              ( '222' )
                                              ( '333' ) ).

  lo_response = http_client->retry_execute(
    i_retry_status_codes = lt_retry_status_codes      " Table
for HTTP status codes for whose entries retry is done
    i_method           = if_web_http_client=>post " HTTP
Method (GET, POST etc.)
    i_idempotent_method = abap_true                 "
Idempotent request method
  ).

  DATA(status) = lo_response->get_status( ).
```

```
IF status-code NE 200.  
    "Error handling here  
ENDIF.  
CATCH cx_web_http_client_error cx_http_dest_provider_error.  
    " handle exception here  
ENDTRY. has occurred!''.  
ENDTRY.
```

### 4.2.9.3.5.2 Set Up HTTP Communication

To set up HTTP communication, use the corresponding communication management apps.

#### Context

The communication management is done by the administrator in SAP Fiori launchpad.

#### Procedure

1. Create a communication system. A communication system determines which target system is called and which authentication methods are used. It also provides the user that is required to register at the target system. For more information, see [Communication Systems \[page 2598\]](#).
2. In the communication system, create an outbound communication user.
3. Create a communication arrangement. The communication arrangement is based on the communication scenario, that is created by the developer in ABAP Development Tools for Eclipse. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

#### Result

The communication management established a connection to the system from which you want to consume the HTTP service. You can now perform the service call as described in [HTTP Communication via Communication Arrangements \[page 994\]](#).

## 4.2.9.3.6 RFC Outbound Communication

RFC outbound connectivity enables you to consume RFC services from other systems. To enable RFC outbound communication, there are two different stages that are performed by different roles:

### Development Tasks

Development tasks are performed by developers. Development tasks are all tasks that are performed within ABAP development tools for Eclipse, such as:

- Creating a Communication Scenario
- Creating a Service Consumption Model (optional, recommended)
- Creating an outbound service

For more information, see [Enable RFC Communication in Your ABAP Code \[page 1004\]](#).

### Administrator Tasks

Administrator tasks are performed by administrators. Administrator tasks are all tasks that are performed in SAP Fiori launchpad, such as:

- Creating a communication system
- Creating a communication arrangement
- Creating communication users

For more information, see [Set Up RFC Communication \[page 1013\]](#).

[Enable RFC Communication in Your ABAP Code \[page 1004\]](#)

[Set Up RFC Communication \[page 1013\]](#)

To set up RFC communication, use the corresponding communication management apps.

## 4.2.9.3.6.1 Enable RFC Communication in Your ABAP Code

### Overview

The RFC client allows to connect to RFC-enabled function modules in remote systems.

#### ⓘ Note

Only synchronous RFC calls are supported.

## Feature Scope

There are two development approaches to provide configuration for RFC communication. Each development approach has different setup options:

- Communication arrangement
  - Provides APIs for receiver determination based on arbitrary properties (see [Define Specific Properties for Communication Scenarios \[page 894\]](#)).
  - Local configuration via communication management UIs or central configuration in SAP BTP cockpit via SAP BTP destination service reference in the communication system (not relevant for Developer Extensibility, see [Developer Extensibility](#)). For more information, see [Create RFC Destinations and Communication Systems \[page 2598\]](#).
  - Requires the development of communication scenario artifacts.
- Destination service (not relevant for Developer Extensibility, see [Developer Extensibility](#)).
  - Provides APIs to fetch connection information for a given destination name from the SAP BTP destination service. No additional development artifacts required.
  - Central configuration in SAP BTP cockpit.

### Note

When using on-premise connectivity, make sure the called RFC function module is exposed in Cloud Connector.

We recommend using the communication arrangement approach for the following reasons:

- There are no hardcoded values/assumptions for a destination name in the destination service or destination service integration (for example, only default integration).
- The communication system can also refer to a destination service. The destination name and destination service instance are then configured instead of hardcoded.

### Note

The communication arrangement approach requires an existing communication scenario and outbound service. See [Communication Systems \[page 2598\]](#) and [RFC Communication via Communication Arrangements \[page 1007\]](#) for more information.

## Service Consumption Model

Other than with OData, SOAP, or HTTP communication, you can either configure an RFC connection with a Service Consumption Model (SRVC) or without SRVC using the standard `CALL FUNCTION ... DESTINATION` statement. For more information, see [Service Consumption Model as RFC Consumer \[page 1006\]](#).

### [Service Consumption Model as RFC Consumer \[page 1006\]](#)

Instead of using the `CALL FUNCTION ... DESTINATION` statement, you can configure an RFC connection with a Service Consumption Model (SRVC).

### [RFC Communication via Communication Arrangements \[page 1007\]](#)

## 4.2.9.3.6.1.1 Service Consumption Model as RFC Consumer

Instead of using the `CALL FUNCTION ... DESTINATION` statement, you can configure an RFC connection with a Service Consumption Model (SRVC).

### Concept

Based on tool support in ABAP development tools for Eclipse (ADT), you can generate an ABAP proxy for calling one or more remote-enabled function modules (RFMs) using the SRVC. To learn how to create a Service Consumption Model with ADT, see [Creating Service Consumption Model](#).

The proxy class contains a specific method for each called RFM. The main benefit of using an SRVC for RFC is that in this class, all data types required for the RFM parameters are generated automatically.

The best way to define an RFC proxy class is to use it for calling only one RFM, or a few RFMs that belong together semantically.

For more information on generating an RFC proxy class based on an SRVC, see [Generating Proxies for Remote Function Call \(RFC\)](#).

In ADT, you can find an example code snippet in the SRVC object of the corresponding RFM.

#### ⓘ Note

We recommend to create a new package for the Service Consumption Model. Generated objects are then created within this package. This allows better organization and clarity.

### Advantages

In many cases, you can reduce development efforts for synchronous RFC calls to other systems significantly by generating an RFC proxy via the SRVC.

The SRVC generates data types required for a typed access to the RFC response.

When using the `CALL FUNCTION ... DESTINATION` statement in these cases, you would have to generate all affected data types manually.

The effort increases with the number and complexity of the required data types. For example, in ABAP applications containing tables with a large number of columns, manual data type generation would be time-consuming. This effort is reduced significantly when using an SRVC.

You can save even more time if you're calling the same RFM in different releases and some of the required data types have different characteristics in each release. In this case, you wouldn't only have to generate the different data types for each release separately, but also in different versions. Note that for each release, one proxy is required.

You can also benefit from RFC calls via SRVC if you want to perform object-oriented development in a consistent manner.

## 4.2.9.3.6.1.2 RFC Communication via Communication Arrangements

To establish communication via RFC, you need to create an outbound service of type RFC and use a Service Consumption Model (SRVC) or the `CALL FUNCTION ... DESTINATION` statement to call other systems from the ABAP environment.

### Prerequisites

- You've created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).
- If you want to call other systems via SRVC, you must have created an SRVC of type RFC as described in [Generating Proxies for Remote Function Call \(RFC\)](#).

### Procedure

1. Create a corresponding outbound service of type RFC.
  1. In your ABAP project, select the relevant package node in the Project Explorer.
  2. Open the context menu and choose `File > New > Other ABAP Repository Object > Cloud Communication Management > Outbound Service > Next` to launch the creation wizard.
  3. Enter the name of the outbound service.
  4. Select RFC in the `Service Type` dropdown list.
  5. Choose `Next` and select a transport request.
  6. Optional: In field `RFC Function Module`, enter the name of the relevant RFC function module.

#### ① Note

You can use this field to document a used RFC function module.

7. Save the outbound service.
2. Add the newly created outbound service to a communication scenario. See [Service Consumption via Communication Arrangements \[page 892\]](#) for more information. According to your developed communication scenario, pass the ID of your developed communication scenario to the `comm_scenario` parameter.

#### ① Note

You can't use the `create_by_comm_arrangement` method for SAP-delivered scenarios.

To call an RFC function module in your communication scenario, it's sufficient to define one outbound service of type RFC that can be used to call multiple RFC function modules. For this outbound service,

go to the corresponding communication scenario. On the *Outbound* tab in the *Outbound Service* section, select [Generate Destination](#).

To document which RFC function modules are called by a communication scenario, you can define separate outbound services and define the name of the function modules without selecting [Generate Destination](#).

3. Call the RFC service as described in the example. Copy the code from the [Overview](#) tab in your SRVC and add the `comm_system_id` and `service_id` parameters if necessary.

## Example

### Using a Service Consumption Model

This example shows how to use RFC communication with an SRVC.

#### ↔ Sample Code

```
DATA: lr_cscn TYPE if_com_scenario_factory=>ty_query-cscn_id_range.

" find Communication Arrangement by scenario ID
lr_cscn = VALUE #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ).
DATA(lo_factory) = cl_com_arrangement_factory->create_instance( ).
lo_factory->query_ca(
    EXPORTING
        is_query = VALUE #( cscn_id_range = lr_cscn )
    IMPORTING
        et_com_arrangement = DATA(lt_ca) ).

IF lt_ca IS INITIAL.
    EXIT.
ENDIF.

" take the first one
READ TABLE lt_ca INTO DATA(lo_ca) INDEX 1.

" get destination based on Communication Arrangement
TRY.
    DATA(lo_dest) = cl_rfc_destination_provider->create_by_comm_arrangement(
        EXPORTING
            comm_scenario = '<Scenario ID>'
        ).
    CATCH cx_rfc_dest_provider_error.
        " handle CX_RFC_DEST_PROVIDER_ERROR
ENDTRY.

" Service Consumption Model:
TRY.
    CREATE OBJECT myobj
        EXPORTING
            destination = lo_dest.
    myobj->bapi_activitytype_getlist(
        IMPORTING
            return           = lt_return
        CHANGING
            activitytype_list = lt_acttype_list.
    ).
    CATCH cx_aco_communication_failure INTO DATA(lcx_comm).
        " handle CX_ACO_COMMUNICATION_FAILURE (sy-msg* in lcx_comm->IF_T100_MESSAGE~T100KEY)
```

```

CATCH cx_aco_system_failure INTO DATA(lcx_sys).
    " handle CX_ACO_SYSTEM_FAILURE (sy-msg* in lcx_sys-
>IF_T100_MESSAGE~T100KEY)
    CATCH cx_aco_application_exception INTO DATA(lcx_appl).
        " handle APPLICATION_EXCEPTIONS (sy-msg* in lcx_appl-
>IF_T100_MESSAGE~T100KEY)
ENDTRY.

```

## ⓘ Note

We recommend to retrieve the correct destination reference based on the communication scenario and a customer-defined property as described in [Service Consumption via Communication Arrangements \[page 892\]](#).

## Using CALL FUNCTION ... DESTINATION

To use the CALL FUNCTION ... DESTINATION statement, use the following code:

### ↗ Sample Code

```

DATA: lr_cscn TYPE if_com_scenario_factory=>ty_query-cscn_id_range.
    " find Communication Arrangement by scenario ID
    lr_cscn = VALUE #( ( sign = 'I' option = 'EQ' low = '<Scenario ID>' ) ).
    DATA(lo_factory) = cl_com_arrangement_factory=>create_instance( ).
    lo_factory->query_ca(
        EXPORTING
            is_query = VALUE #( cscn_id_range = lr_cscn )
        IMPORTING
            et_com_arrangement = DATA(lt_ca) .
    IF lt_ca IS INITIAL.
        EXIT.
    ENDIF.
    " take the first one
    READ TABLE lt_ca INTO DATA(lo_ca) INDEX 1.
    " get destination based on Communication Arrangement
    TRY.
        DATA(lo_dest) =
        cl_rfc_destination_provider=>create_by_comm_arrangement(
            EXPORTING
                comm_scenario = '<Scneario ID>'
                service_id = '<Outbound Service ID>'
                comm_system_id = lo_ca->get_comm_system_id( )
            ).
        DATA(destination_name) = lo_dest->get_destination_name( ) .
        CATCH cx_rfc_dest_provider_error.
            " handle CX_RFC_DEST_PROVIDER_ERROR
    ENDTRY.
    " CALL FUNCTION
    DATA msg TYPE c LENGTH 255.
    CALL FUNCTION 'BAPI_ACTIVITYTYPE_GETLIST' DESTINATION destination_name
        IMPORTING
            return
                = lt_return
        TABLES
            activitytype_list
                = lt_acttype_list
        EXCEPTIONS
            system_failure
                = 1 MESSAGE msg
            communication_failure
                = 2 MESSAGE msg
            OTHERS
                = 3.
    CASE sy-subrc.
        WHEN 1.
            " handle system failure
        WHEN 2.
            " handle communication failure
        WHEN 3.

```

```
    " handle application failure  
ENDCASE.
```

## Test Your Outbound Call

To test your outbound call, you have to provide a configuration for the outbound service you want to call in your code.

Create a communication system and communication arrangement for the communication scenario in the Communication Systems and Communication Arrangements apps and maintain the required data.

## Related Information

[Service Consumption via Communication Arrangements \[page 892\]](#)

[Call a Remote Function Module \(RFC\) From SAP BTP, ABAP Environment](#) 

## 4.2.9.3.6.1.3 RFC Communication via Destination Service (Deprecated)

### Context

Use the destination service in SAP BTP to store destination information that can be reused by SAP BTP app services.

#### Note

You can use the destination service. However, this approach is deprecated. We recommend using the communication arrangement approach instead. See [RFC Communication via Communication Arrangements \[page 1007\]](#) for more information.

You've the following options to consume a destination service:

- Dynamic by using a communication arrangement and a communication system
- Static by using the method `create_by_cloud_destination` in the code

## Prerequisites

If you want to call other systems via SRVC, you must have created an SRVC of type RFC as described in [Generating Proxies for Remote Function Call \(RFC\)](#).

## Procedure

When using `create_by_cloud_destination`, proceed as follows:

Create a destination object using class `cl_http_destination_provider` and method `create_by_cloud_destination` with the following parameters:

- `i_name`: the name of the destination.
- Optional: `i_service_instance_name`: Typically, you use the destinations of the subaccount in which the ABAP instance resides or, in case of a SaaS solution based on the ABAP environment, the destinations of the consumer subaccount. However, you can add more destinations using your own destination service instance and communication scenario SAP\_COM\_0276, for example, to achieve separation of concerns (see also [Create a Destination \[page 891\]](#)). In this case, specify the value of the service instance name property of the communication arrangement for SAP\_COM\_0276.

## Example

### Using a Service Consumption Model

You can consume your service using a Service Consumption Model (SRVC). Create an SRVC and replace the call of the method `cl_rfc_destination_provider=>create_by_comm_arrangement( )` that is proposed in the SRVC with `cl_rfc_destination_provider=>create_by_cloud_destination( )`:

#### ↔ Sample Code

```
TRY.  
  
    " replace Method create_by_comm_arrangement( ).  
    " dest = cl_rfc_destination_provider=>create_by_comm_arrangement(  
    " comm_scenario = 'MY_COMM_SCENARIO'  
    " ).  
  
    " replacement  
    DATA(dest) = cl_rfc_destination_provider=>create_by_cloud_destination(  
    EXPORTING  
        i_name      = 'name of destination'  
    ).  
  
        CREATE OBJECT myobj  
        EXPORTING  
            destination = dest.  
  
        myobj->bapi_actuitytype_getlist(  
        IMPORTING  
            return      = lt_return  
        CHANGING  
            activitytype_list = lt_acttype_list.
```

```

).
  CATCH cx_aco_communication_failure INTO DATA(lcx_comm).
    " handle CX_ACO_COMMUNICATION_FAILURE (sy-msg* in lcx_comm-
>IF_T100_MESSAGE~T100KEY)
    CATCH cx_aco_system_failure INTO DATA(lcx_sys).
      " handle CX_ACO_SYSTEM_FAILURE (sy-msg* in lcx_sys-
>IF_T100_MESSAGE~T100KEY)
    CATCH cx_aco_application_exception INTO DATA(lcx_appl).
      " handle APPLICATION_EXCEPTIONS (sy-msg* in lcx_appl-
>IF_T100_MESSAGE~T100KEY)
    CATCH cx_rfc_dest_provider_error.
      " handle CX_RFC_DEST_PROVIDER_ERROR

ENDTRY.

```

### Using CALL FUNCTION ... DESTINATION

The sample code below shows how to get a reference to a destination and how to use it when calling a remote-enabled function module.

#### Sample Code

```

" getting the reference to the relevant destination
TRY.
DATA(lr_dest) = cl_rfc_destination_provider->create_by_cloud_destination(
  EXPORTING
    i_name          = 'name_of_destination'
).
CATCH cx_rfc_dest_provider_error.
  " handle CX_RFC_DEST_PROVIDER_ERROR
ENDTRY.

DATA(lo_dest) = lr_dest->get_destination_name( ).
" using the destination in RFC

DATA msg TYPE c LENGTH 255.
  CALL FUNCTION 'BAPI_ACTIVITYTYPE_GETLIST' DESTINATION lo_dest-
>get_destination_name( )
  IMPORTING
    return          = lt_return
  TABLES
    activitytype_list = lt_acttype_list.
EXCEPTIONS
  system_failure      = 1 MESSAGE msg
  communication_failure = 2 MESSAGE msg
  OTHERS              = 3.
CASE sy-subrc.
  WHEN 1.
    " handle system failure
  WHEN 2.
    " handle communication failure
  WHEN 3.
    " handle application failure
ENDCASE.

```

#### Note

When using on-premise connectivity, make sure the called RFC function module is exposed in Cloud Connector. See [Configure Access Control \(RFC\)](#).

## Test Your Outbound Call

To test your outbound call, configure an RFC destination as described in [Create RFC Destinations](#). The following authentication methods are supported in the SAP BTP, ABAP environment:

### Internet:

- Basic Authentication: Fill the *User* and *Password* fields. If an alias logon is required, use the field *Alias User*.
- Client Certificate Authentication: Use the additional property `jco.client.tls_client_certificate_logon` with value `1` to enable Client Certificate Authentication.

### OnPremise:

- Basic Authentication: Fill the *User* and *Password* fields.
- Principal Propagation: Use the additional property `jco.destination.auth_type` with value `PrincipalPropagation` to enable Principal Propagation.

#### ⓘ Note

- For proxy type *Internet* and the use of authentication type client certificate, you must upload the X.509 client certificate in P12 format on the client side in the SAP BTP, ABAP environment, using the Maintain Client Certificates application. For more information, see [Maintain Client Certificates \[page 2772\]](#). Uploading the client certificate via destination service isn't supported.
- For proxy type *OnPremise*, the use of authentication type `PrincipalPropagation` for a destination isn't supported in the ADT class runner.

## Related Information

[Destination Service \[page 885\]](#)

## 4.2.9.3.6.2 Set Up RFC Communication

To set up RFC communication, use the corresponding communication management apps.

## Context

The communication management is done by the administrator in SAP Fiori launchpad.

#### ⓘ Note

If you've configured on-premise connectivity (via Cloud Connector) in the RFC settings of your communication system or destination service, use fast serialization. See SAP Note [2418683](#) for more

information. For example, if you use a Service Consumption Model, data and structures containing includes can be transferred incorrectly because of displaced offsets. Fast serialization is set by default.

## Procedure

1. Create a communication system. A communication system determines which target system is called and which authentication methods are used. It also provides the user that is required to register at the target system. For more information, see [Communication Systems \[page 2598\]](#).
2. In the communication system, create an outbound communication user.
3. Create a communication arrangement. The communication arrangement is based on the communication scenario, that is created by the developer in ABAP Development Tools for Eclipse. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

## Result

The communication management established a connection to the system from which you want to consume the RFC service. You can now perform the service call as described in [RFC Communication via Communication Arrangements \[page 1007\]](#).

### 4.2.9.3.7 SOAP Outbound Communication

Learn how to set up and enable SOAP outbound communication.

SOAP outbound connectivity enables you to consume web services from other systems. To enable SOAP outbound communication, there are three different stages that are performed by different roles:

## Development Tasks

Development tasks are performed by developers. Development tasks are all tasks that are performed within ABAP development tools for Eclipse, such as:

- Creating a Communication Scenario
- Creating a Service Consumption Model
- Creating an outbound service

For more information, see [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#).

## Administrator Tasks

Administrator tasks are performed by administrators. Administrator tasks are all tasks that are performed in SAP Fiori launchpad, such as:

- Creating a communication system
- Creating a communication arrangement
- Creating communication users

For more information, see [Set Up SOAP Communication \[page 1030\]](#).

## Monitoring Tasks

Monitoring tasks are performed by administrators. Monitoring tasks are all tasks that are performed in the respective monitoring apps in SAP Fiori launchpad, such as:

- SOAP error log
- Message Monitoring

For more information, see [SOAP Monitoring \[page 1031\]](#).

[Enable SOAP Communication in Your ABAP Code \[page 1015\]](#)

[Set Up SOAP Communication \[page 1030\]](#)

To set up SOAP communication, use the corresponding communication management apps.

[SOAP Monitoring \[page 1031\]](#)

### 4.2.9.3.7.1 Enable SOAP Communication in Your ABAP Code

#### Feature Scope

SOAP-based web services cover the following features:

- Synchronous and asynchronous outbound web service communication
- Self-contained WSDL (single file with no import or include statements)
- gCTS Support (not relevant for Developer Extensibility, see [Developer Extensibility](#))
- Web service protocols, e.g. to add custom XML elements to the SOAP request header, add the Application Interface Key, or enable EOIO processing. For more information, see [SOAP Protocols \[page 1024\]](#).

To enable SOAP communication, there are three different approaches:

- Communication arrangement
  - Provides APIs for receiver determination based on arbitrary properties (see [Define Specific Properties for Communication Scenarios \[page 894\]](#) ).

- Local configuration via communication management UIs or central configuration in SAP BTP cockpit via SAP BTP destination service reference in the communication system (not relevant for Developer Extensibility, see [Developer Extensibility](#)). For more information, see [Communication Systems \[page 2598\]](#).
- Requires the development of communication scenario artifacts.
- Destination service (not relevant for Developer Extensibility, see [Developer Extensibility](#)).
  - Provides APIs to fetch connection information for a given destination name from the SAP BTP destination service. No additional development artifacts required.
  - Central configuration in SAP BTP cockpit.
- URL approach: By using a plain URL and setter methods to provide the needed configuration directly in the coding.

All of the listed approaches above require a Service Consumption Model. For more information, see [Service Consumption Model as SOAP Consumer \[page 1018\]](#).

#### 4.2.9.3.7.1.1 Enhanced WSDL Document

The enhanced Web Services Description Language (WSDL) document is a WSDL document enhanced by ABAP information that is relevant for design time.

When you use a regular WSDL as input to create a Service Consumption Model (SRVC), the design time relevant information, such as the types, messages, and port type sections, are analyzed and interpreted by the ABAP Web service framework. For all the ABAP properties like the names and types, it will make suggestions and generate a WSDL similar to the input WSDL enhanced with this ABAP information. If needed, you can change these ABAP properties within the enhanced WSDL document to adjust the ABAP artifacts accordingly (upon activation). The mapping between XSD and ABAP types in particular can be adjusted this way. Of course, an enhanced WSDL document can be used as input as well.

To encode ABAP properties into the WSDL, the namespace `http://sap.com/abap/proxy` with the prefix `abap` was introduced: `xmlns:abap=http://sap.com/abap/proxy`. For any applicable node of a WSDL, the corresponding enhanced WSDL document contains additional nodes within this namespace to accommodate its ABAP properties. Wherever you see the `abap` prefix in the WSDL, you can adjust the ABAP property. These additional nodes can be XSD attributes or elements, depending on where the node to be enhanced is located. This is necessary to maintain WSDL validity. Schema nodes such as `xmlns:xsd=http://www.w3.org/2001/XMLSchema` are enhanced via additional XSD attributes. WSDL nodes `xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/`, on the other hand, are enhanced via additional XSD (sub)elements. Regular ABAP properties of any node contain the following:

- Prefix
- Name
- Description
- ABAP type

With the ABAP type, you can control the mapping between XSD and ABAP types. The property `availableTypes` displays all applicable ABAP types in this case. To change the ABAP type, copy and paste one of the listed types into the `type` attribute of the given element. Afterwards, reactivate the SRVC to apply the changes.

## Summary of Additional Nodes and Their Technical Types

Node name	Technical Type	Description
abap:prefix	Up to 20 characters, case insensitive	ABAP prefix (only used if the object is represented by an ABAP object such as CLAS, INTF, TABL, DTEL)
abap:name	Up to 30 characters, case insensitive	ABAP name
abap:type	String	Default or set ABAP type
abap:availableTypes	Comma-separated list of 30 characters	List of all available ABAP types
abap:description	Up to 60 characters	Description
abap:prefixT	See abap:prefix	Prefix for corresponding table type (if maxOccurs > 1 and additional table type is generated and a prefix is required)
abap:nameT	See abap:name	Name for corresponding table type (if maxOccurs > 1 and additional table type is generated)
abap:descriptionT	See abap:description	Description for corresponding table type (if maxOccurs > 1 and additional table type is generated)

### Example

The following code snippet displays a types section of an enhanced WSDL with a schema node (prefix xsd), where all ABAP properties are encoded as XSD attributes:

#### Sample Code

```

<xsd:element name="NumberToWords">
  <xsd:complexType abap:prefix="ZZZ_" abap:name="ZZZ_NUMBER_TO_WORDS_SOAP_REQUE" abap:description="Proxy Structure (generated)">
    <xsd:sequence>
      <xsd:element name="ubiNum" form="qualified" type="xsd:unsignedLong" abap:name="UBI_NUM" abap:type="INT8" abap:availableTypes="INT8, INT4, DEC-20"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>

```

The element NumberToWords being a complexType is represented by the dictionary structure ZZZ\_NUMBER\_TO\_WORDS\_SOAP\_REQUE in ABAP. The element within this complexType ubiNum of XSD type xsd:unsignedLong corresponds to structure entry UBI\_NUM of ABAP type INT8, which you could change to INT4 or DEC-20 (availableTypes).

The second snippet represents an exemplary portType section, where all ABAP properties are encoded as additional XSD elements (prefix wsdl1) rather than attributes:

#### Sample Code

```

<wsdl1:portType name="NumberConversionSoapType">
  <abap:prefix>ZZZ_</abap:prefix>

```

```

<abap:name>ZZZ_CO_NUMBER_CONVERSION_SOAP</abap:name>
<abap:description>Proxy Class (generated)</abap:description>
<wsdl:operation name="NumberToWords">
  <abap:name>NUMBER_TO_WORDS</abap:name>
  <abap:description>Returns the word corresponding to the positive number
passed</abap:description>
  <wsdl:input message="tns:NumberToWordsSoapRequest" />
  <wsdl:output message="tns:NumberToWordsSoapResponse" />
</wsdl:operation>
</wsdl:portType>

```

In this code snippet, the generated ABAP class is called `ZZZ_CO_NUMBER_CONVERSION_SOAP` and its method is `NUMBER_TO_WORDS`, which corresponds to the operation `NumberToWords` of the service.

### 4.2.9.3.7.1.2 Service Consumption Model as SOAP Consumer

A Service Consumption Model is the main requirement for consuming a web service in the ABAP environment. To learn how to create a Service Consumption Model with ABAP development tools for Eclipse (ADT), see [Creating Service Consumption Model](#).

#### Note

We recommend to create a new package for the Service Consumption Model. Generated objects are then created within this package. This allows better organization and clarity.

For the consumption type web service, you need to upload the WSDL of the service you want to consume. See [Generating Proxies for Remote Web Service](#). Upon successful activation, all the dependent objects are created and displayed in the ADT project explorer, such as:

- The enterprise service
- Corresponding dictionary objects
- The generated ABAP class

For any operation of the web service, a code snippet is displayed in ADT that indicates how to call the service via the corresponding method of the generated class. It is instantiated using a destination object that refers to a communication arrangement and scenario.

An enhanced WSDL document has been introduced that describes how schema/WSDL entities are mapped to ABAP. In the WSDL, any schema element is enhanced by ABAP properties such as ABAP name and type. You can change all these ABAP properties by editing the enhanced WSDL and reactivating the Service Consumption Model. To learn more about the enhanced WSDL, see [Enhanced WSDL Document \[page 1016\]](#).

#### Note

The Service Consumption Model is initially saved in an inactive state. Therefore, not all dependent objects are immediately visible in the project explorer. Upon activation, all dependent objects are created according to the enhanced WSDL. They are then visible in the project explorer.

## 4.2.9.3.7.1.3 SOAP Communication via Communication Arrangements

### Prerequisites

- You have created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).
- You have a service meta data file (WSDL file) for the service you want to consume.
- You have created and activated a service consumption model (SRVC) of type `Web Service`. See [Generating Proxies for Remote Web Service](#) for more information.

#### ⚠ Restriction

When calling a SOAP service via communication arrangements, only SOAP 1.1 is supported.

### Procedure

1. Create a corresponding outbound service of type SOAP.
  1. In your ABAP project, select the relevant package node in the Project Explorer.
  2. Open the context menu and choose  `File`  `New`  `Other ABAP Repository Object`  `Cloud Communication Management`  `Outbound Service`  `Next` to launch the creation wizard.
  3. Enter the name of the outbound service.
  4. Select `SOAP` in the `Service Type` dropdown list.
  5. Choose `Next` and select a transport request.
  6. In field `Service Interface`, enter the name of the relevant consumer proxy. You can choose the consumer proxy from the list of the value help.
  7. Save the outbound service.
2. Add the newly created outbound service to the communication scenario. See [Service Consumption via Communication Arrangements \[page 892\]](#) for more information.

#### ⓘ Note

You can't use the `create_by_comm_arrangement` method for SAP-delivered scenarios.

3. Publish the communication scenario. The administrator can then create the required communication management objects as described in [Test Your Outbound Call \[page 1021\]](#).
4. Call the SOAP service as described in the example. Copy the code snippet from the `Overview` tab in your SRVC and add the `comm_system_id` and `service_id` parameters if necessary. Note the difference between synchronous and asynchronous services. According to your developed communication scenario, add the following:

comm_scenario	mandatory	ID of the developed communication scenario
comm_system_id	optional	ID of the configured communication system
service_id	optional	ID of the developed outbound service

## Example

### Synchronous Services

#### ↔ Sample Code

```

TRY.
  DATA(soap_destination) =
    cl_soap_destination_provider->create_by_comm_arrangement(
      comm_scenario = '<demo scenario>'
      service_id   = '<service id>'
      comm_system_id = '<comm system>' ).

  DATA(proxy) = NEW zsc_co_epm_product_soap( destination =
    soap_destination ).

  DATA(request) = VALUE zsc_req_msg_type( req_msg_type-product = '<product
name>' ).
  proxy->get_price(
    EXPORTING
      input = request
    IMPORTING
      output = DATA(response) .

  "handle response
  CATCH cx_soap_destination_error.
    "handle error
  CATCH cx_ai_system_fault.
    "handle error
  CATCH zsc_cx_fault_msg_type.
    "handle error

ENDTRY.

```

#### ⓘ Note

We recommend to retrieve the correct destination reference based on the communication scenario and a customer-defined property as described in [Service Consumption via Communication Arrangements \[page 892\]](#).

### Asynchronous Services

For asynchronous services, follow the same procedure as for synchronous services. The only differences in the code are the following:

- The `IMPORTING` parameter is missing since the service doesn't return any value.
- A `COMMIT WORK` triggers the SOAP call.

## ↔ Sample Code

```
TRY.  
    DATA(destination) =  
cl_soap_destination_provider->create_by_comm_arrangement(  
    comm_scenario = '<demo scenario>'  
    *           service_id   = '<service id>'  
    *           comm_system_id = '<comm system>'  
    ).  
    DATA(proxy) = NEW zco_appointment_activity_reque(  
        destination = destination  
    ).  
    DATA(request) = VALUE zappointment_activity_request1( ).  
    proxy->appointment_activity_request_i(  
        EXPORTING  
            input = request  
    ).  
    COMMIT WORK. "to trigger async call  
CATCH cx_soap_destination_error.  
    "handle error  
CATCH cx_ai_system_fault.  
    "handle error  
ENDTRY.
```

## Test Your Outbound Call

To test your outbound call, you have to provide a configuration for the outbound service you want to call in your code.

Create a communication system and communication arrangement for the communication scenario in the Communication Systems and Communication Arrangements apps and maintain the required data.

To check if an asynchronous call was successfully sent to the provider system, see [SOAP Monitoring \[page 1031\]](#).

These tasks are performed by the administrator. For more information, see [Set Up SOAP Communication \[page 1030\]](#) and [SOAP Monitoring \[page 1031\]](#).

## Related Information

[Enable SOAP Communication in Your ABAP Code \[page 1015\]](#)

[Communication Management \[page 883\]](#)

[Communication Arrangement \[page 885\]](#)

[Developing External Service Consumption \(Outbound Communication\) \[page 890\]](#)

## 4.2.9.3.7.1.4 SOAP Communication via Destination Service (Deprecated)

To configure a SOAP-specific destination, create an HTTP destination as described in [Set Up HTTP Communication \[page 1003\]](#).

### ⓘ Note

You can use the destination service. However, this approach is deprecated. We recommend using the communication arrangement approach instead. See [SOAP Communication via Communication Arrangements \[page 1019\]](#) for more information.

You can set Web service-specific properties by maintaining the following *Additional Properties* in the destination:

- `ws.soapVersion`: Sets the SOAP version.
- `ws.maxWaitTime`: Sets the maximum waiting time for the consumer (in seconds).
- `ws.compressMessage`: Enables compression of the message.
- `ws.soapAction.<operationName>`: Sets the SOAP action for a given operation with name `<operationName>`.

### ↗ Sample Code

```
TRY.  
    DATA(lo_soap_dest) =  
        cl_soap_destination_provider->create_by_cloud_destination(  
            i_name                  = '<destination name>'  
            i_service_instance_name  = '<destination service  
instance name>').  
    DATA(proxy) = NEW zco_appointment_activity_reque(  
        destination = destination ).  
    DATA(request) = VALUE zsc_req_msg_type(  
        req_msg_type-product = '<product name>' ).  
    proxy->get_price(  
        EXPORTING  
            input = request  
        IMPORTING  
            output = DATA(response) ).  
  
    "handle response  
    CATCH cx_soap_destination_error.  
    "handle error  
    CATCH cx_ai_system_fault.  
    "handle error  
    CATCH zsc_cx_fault_msg_type.  
    "handle error  
  
ENDTRY.
```

## Related Information

[Enable SOAP Communication in Your ABAP Code \[page 1015\]](#)

## 4.2.9.3.7.1.5 SOAP Communication via URL

Instead of setting the *Additional Properties* as in the destination service approach, you can set them programmatically:

- `set_soap_version( )`: Sets the SOAP version. Valid values for the SOAP version are provided by the constants `if_soap_destination=>soap_version_11` and `if_soap_destination=>soap_version_12`.
- `set_max_wait_time( )`: Sets the maximum waiting time for the consumer (in seconds).
- `set_compress_message( )`: Enables compression of messages if parameter `i_compress_message` is set to `true`. Disables compression if the parameter is set to `false`.
- `set_soap_action( )`: Sets the SOAP action for a given operation. Parameters are the name of the operation and the value of the SOAP action.
- `set_url( )`: Sets the URL of the endpoint.
- `set_basic_authentication( )`: Sets the authentication method to basic authentication and sets user and password.
- `use_client_certificate( )`: Sets the authentication method to client certificate authentication. The default client certificate is used.

### ⓘ Note

An existing service consumption model (SRVC) is required.

### ⓘ Note

Using `create_by_url` is only suitable for public services or test purposes, because credentials should be stored in the communication management in SAP Fiori launchpad.

Call the SOAP service as described in the example. As described in [SOAP Communication via Communication Arrangements \[page 1019\]](#), copy the code snippet from the *Overview* tab in your SRVC. Replace the `CREATE_BY_COMM_ARRANGEMENT` method with the `CREATE_BY_URL` method and pass the required URL to the method.

### ↗ Sample Code

```
TRY.  
"replace create_by_comm_arrangement with create_by_url  
    DATA(soap_destination) = cl_soap_destination_provider=>create_by_url(  
        'https://<host>/sap/bc/srt/xip/sap/<provider>/000/  
<service>/<binding>').  
  
    soap_destination->use_client_certificate( ).  
"generated code snippet  
    DATA(proxy) = NEW zsc_co_service( destination = soap_destination ).  
  
    DATA(request) = VALUE zsc_req_msg_type( req_msg_type-product = '<product  
name>' ).  
        proxy->get_price(  
            EXPORTING  
                input = request  
            IMPORTING  
                output = DATA(response) ).  
  
    "handle response  
    CATCH cx_soap_destination_error.
```

```
    "handle error
    CATCH cx_ai_system_fault.
    "handle error
    CATCH zsc_cx_fault_msg_type.
    "handle error

ENDTRY.
```

## Related Information

[Enable SOAP Communication in Your ABAP Code \[page 1015\]](#)

[Tutorial: Consume SOAP-Based Web Services with SAP BTP ABAP Environment](#) 

### 4.2.9.3.7.1.6 SOAP Protocols

Protocols allow you to use additional SOAP runtime features.

Class `CL_WS_PROTOCOL_FACTORY` provides factory methods that return an interface specific to each protocol that you want to use.

The exception `CX_WS_PROTOCOL_ERROR` is raised whenever a protocol error occurs.

The following protocols are available:

[Add Custom SOAP Header Elements \[page 1025\]](#)

With the web services SOAP header protocol, you can programmatically add custom XML elements to the SOAP request header.

[Add the AIR Key to a SOAP Request in the HTTP Header \[page 1027\]](#)

With the web services HTTP header protocol, you can programmatically add the Application Interface Key (AIR key) field to the HTTP header of a SOAP request.

[Enable EOIO Processing \[page 1029\]](#)

With the web services sequence protocol, you can enable exactly once in order (EOIO) processing for asynchronous web services.

## 4.2.9.3.7.1.6.1 Add Custom SOAP Header Elements

With the web services SOAP header protocol, you can programmatically add custom XML elements to the SOAP request header.

### Context

Some services can require setting specific XML elements in the SOAP header. To add such custom elements, you can use the method `ADD_SOAP_HEADER_ELEMENT`.

### Procedure

1. Create a consumer proxy by instantiating the generated class with a SOAP destination object. See [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#) for more information.
2. Pass the proxy object to the factory method  
`cl_ws_protocol_factory->get_soap_header_protocol( )`. The factory method returns a (proxy-specific) WS protocol object to adjust the SOAP header.
3. Pass the custom XML element as string to the method `add_soap_header_element`.

```
DATA(ws_soap_header_facade) =  
cl_ws_protocol_factory->get_soap_header_protocol( proxy ).  
ws_soap_header_facade->add_soap_header_element( '<XML element>' ).
```

#### ⓘ Note

To add multiple XML elements, call the method for each element separately.

### Exception Handling

The API raises the exception `CX_WS_PROTOCOL_ERROR`, for example, if the code snippet is syntactically incorrect or if one of the following reserved namespaces is used:

- `http://schemas.xmlsoap.org/ws/2004/08/addressing`
- `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd`
- `http://www.w3.org/2005/08/addressing`
- `http://www.sap.com/webas/640/soap/features/messageId`
- `http://schemas.xmlsoap.org/ws/2005/02/rm`
- `http://docs.oasis-open.org/ws-rx/wsrm/200702`
- `http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512`
- `http://schemas.xmlsoap.org/ws/2005/02/sc`

- <http://www.sap.com/webas/630/soap/runtime/session/protocol>

## Example

The following example shows how to adjust the SOAP request header followed by the web service call.

### ⓘ Note

First, you must get a SOAP destination object according to your SAP product as described in [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#).

### ↗ Sample Code

```

" Get destination object
TRY.
  DATA(proxy) = NEW example_consumer( destination = soap_destination ).

  DATA xml_element TYPE string VALUE
    '<n0:CustomHeaderElement xmlns:n0="http://ws.example.org/ws/">' &
    '<n0:CustomElement>' &
    '<n0:value>specific info</n0:value>' &
    '</n0:CustomElement>' &
    '</n0:CustomHeaderElement>' .

  TRY.
    " Pass the proxy object to factory method and add XML element to the
    SOAP request header.
    DATA(ws_header_facade) =
      cl_ws_protocol_factory->get_soap_header_protocol( proxy ).
      ws_header_facade->add_soap_header_element( xml_element ).

    CATCH cx_ws_protocol_error.
      " Handle error.
    ENDTRY.

  " Call server
  DATA(request) = VALUE example_req_msg_type( ).
  proxy->example_operation(
    EXPORTING
      input = request
    IMPORTING
      output = DATA(response) ).

  CATCH cx_ai_system_fault.
    " Handle error.
  ENDTRY.

```

The resulting SOAP request looks like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    ...
    <n0:CustomHeaderElement xmlns:n0="http://ws.example.org/ws/">
      <n0:CustomElement>
        <n0:value>specific info</n0:value>
      </n0:CustomElement>
    </n0:CustomHeaderElement>
  
```

```
...  
</env:Header>  
<env:Body>  
...  
</env:Body>  
</env:Envelope>
```

## Related Information

[Set Up SOAP Communication \[page 1030\]](#)

### 4.2.9.3.7.1.6.2 Add the AIR Key to a SOAP Request in the HTTP Header

With the web services HTTP header protocol, you can programmatically add the Application Interface Key (AIR key) field to the HTTP header of a SOAP request.

## Context

If your application calls an SAP SOAP API, the Application Interface Key must be part of the request. The Application Interface Key is passed in the HTTP header. For SOAP requests, you can do this with the HTTP header protocol. To add this field, you can use the method `SET_FIELD` of the interface `IF_WS_HTTP_HEADER_FACADE`.

#### ⓘ Note

- It isn't possible to add a field other than the Application Interface Key field.
- Once a web service call is performed, the custom HTTP header is reset. If you want to use the same value again, you must set the value again.

## Procedure

1. Create a consumer proxy by instantiating the generated class with a SOAP destination object. See [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#) for more information.
2. Pass the proxy object to the factory method `cl_ws_protocol_factory->get_http_header_protocol( )`. The factory method returns a (proxy-specific) WS protocol object to adjust the HTTP header.
3. Pass the name of the HTTP header field and value as string to the method `set_field`.

## ↔ Sample Code

```
DATA(header) = cl_ws_protocol_factory->get_http_header_protocol( io_proxy
= proxy ).
header->set_field( iv_name =
if_ws_http_header_facade->co_field_appl_interface_key iv_value =
'0123abcd' ).
```

## Example

### ⓘ Note

After executing a service call, the field value is initialized by the SOAP runtime. Therefore, you have to set the HTTP field values for each service call with the `SET_FIELD` method.

## ↔ Sample Code

```
" Get destination object
TRY.
  DATA(destination) =
cl_soap_destination_provider->create_by_comm_arrangement(
  comm_scenario = 'HTTP_HEADER' .
  DATA(proxy) = NEW zco_srt_test_provider( destination = destination ).
  " Pass the proxy object to factory method to obtain an instance of the HTTP
header protocol
  DATA(header) = cl_ws_protocol_factory->get_http_header_protocol( io_proxy
= proxy ).
  " First service call
  " Add HTTP field to the SOAP request.
  header->set_field( iv_name =
if_ws_http_header_facade->co_field_appl_interface_key iv_value = '0123abcd' ) .
  " Call service
  DATA(request) = VALUE zrfc_system_info( ).
  proxy->rfc_system_info( EXPORTING input = request IMPORTING output =
DATA(response) ) .
  " Second service call
  " Add HTTP field to the SOAP request.
  header->set_field( iv_name =
if_ws_http_header_facade->co_field_appl_interface_key iv_value = '9876wxyz' ) .
  " Call service
  DATA(request) = VALUE zrfc_system_info( ).
  proxy->rfc_system_info( EXPORTING input = request IMPORTING output =
DATA(response) ) .
  CATCH cx_ws_protocol_error INTO DATA(lx_protocol_error).
    " Handle protocol error
  CATCH cx_soap_destination_error INTO DATA(lx_error).
    " Handle destination error
  CATCH cx_ai_system_fault INTO DATA(lx_fault).
    " Handle system fault
ENDTRY.
```

## 4.2.9.3.7.1.6.3 Enable EOIO Processing

With the web services sequence protocol, you can enable exactly once in order (EOIO) processing for asynchronous web services.

### Context

It can sometimes be important that service requests arrive at the provider side in the correct order. This is the case when database entries that depend on a previous write process are being written. If a database entry is changed that has not yet even been created, errors occur.

To ensure that calls are processed in the correct order, calls that belong together are given a sequence and then are evaluated by the provider.

If proxy object methods are called as part of a sequence, these calls are processed 'exactly once in order' (EOIO). If the proxy object methods are called without sequence, the methods are called 'exactly once' without a particular order.

To enable EOIO processing, proceed as follows.

### Procedure

1. Create a consumer proxy by instantiating the generated class with a SOAP destination object. See [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#) for more information.
2. Pass the proxy object to the factory method `cl_ws_protocol_factory->get_sequence_protocol( )`. The factory method returns a (proxy-specific) WS protocol object.
3. To enable EOIO processing, use the methods `ws_sequence_facade->begin_eoio_sequence( )` and `ws_sequence_facade->end_eoio_sequence( )`. The proxy calls in between these two methods are processed EOIO.
4. Use `COMMIT WORK` to trigger web service calls in the EOIO sequence block.

#### Sample Code

```
DATA(ws_sequence_facade) =  
  cl_ws_protocol_factory->get_sequence_protocol( proxy ).  
  ws_sequence_facade->begin_eoio_sequence( ).  
    <WS calls to be executed EOIO>  
  ws_sequence_facade->end_eoio_sequence( ).
```

#### Note

If there are multiple `begin/end_eoio_sequence` blocks, the corresponding sequences are executed independently of each other. All messages within a sequence are still executed exactly once in order.

## Example

### ⓘ Note

First, you must get a SOAP destination object according to your SAP product as described in [Enable SOAP Communication in Your ABAP Code \[page 1015\]](#).

### ↴ Sample Code

```
" Get destination object
TRY.
    DATA(proxy) = NEW zco_appointment_activity_reque(
        destination = destination
    ).

    DATA(ws_sequence_facade) =
cl_ws_protocol_factory->get_sequence_protocol( proxy ).

    " Begin EOIO sequence
    ws_sequence_facade->begin_eoio_sequence( ).
    DATA(request1) = VALUE zco_appointment_activity_reque( ).
    proxy->check_operation(
        EXPORTING
            input = request1
    ).
    DATA(request2) = VALUE zco_appointment_activity_reque( ).
    proxy->check_operation(
        EXPORTING
            input = request2
    ).
    " End EOIO sequence.
    ws_sequence_facade->end_eoio_sequence( ).

    COMMIT WORK.

    CATCH cx_ws_protocol_error
        "handle response
    CATCH cx_ai_system_fault.
        "handle error
    CATCH cx_srt_check_oper_fault.
        "handle error
ENDTRY.
```

## 4.2.9.3.7.2 Set Up SOAP Communication

To set up SOAP communication, use the corresponding communication management apps.

### Context

The communication management is done by the administrator in SAP Fiori launchpad.

## Procedure

1. Create a communication system. A communication system determines which target system is called and which authentication methods are used. It also provides the user that is required to register at the target system. For more information, see [Communication Systems \[page 2598\]](#).
2. In the communication system, create an outbound communication user.
3. Create a communication arrangement. The communication arrangement is based on the communication scenario, that is created by the developer in ABAP Development Tools for Eclipse. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

## Result

The communication management established a connection to the system from which you want to consume the SOAP service. You can now perform the service call as described in [SOAP Communication via Communication Arrangements \[page 1019\]](#).

### 4.2.9.3.7.3 SOAP Monitoring

#### Prerequisites

You can find error logs related to SOAP services in the SOAP Error Log application. For more information, see [SOAP Error Log, OData Error Log, and Event Error Log \[page 2713\]](#)

To monitor asynchronous SOAP calls, use the Message Monitoring Overview app. To use the app, the following is required.

- Roles based on the following business catalogs are assigned to your user in SAP Fiori launchpad:
    - SAP\_CA\_BC\_COM\_ERR\_PC
    - SAP\_CA\_BC\_COM\_CONF\_PC
- See [Authorization Basics \[page 805\]](#) and [Business Catalogs \[page 2699\]](#) for more information.
- The user is a recipient of the /SPRX namespace. You can manage this in the Assign Recipients to Users app. See [Assign Recipients to Users \[page 2711\]](#) for more information.

#### Context

**Asynchronous SOAP services:** When calling asynchronous SOAP services as described in [SOAP Communication via Communication Arrangements \[page 1019\]](#), you can check if the call has been sent to the receiver using the Message Monitor Overview app. See [Message Monitoring Overview \[page 2709\]](#) for more information.

## Procedure

1. Create a Service Consumption Model (SRVC) of type `Web Service` as described in [Creating Service Consumption Model](#).
2. Activate the SRVC. This creates a so-called AIF interface in the Message Monitor Overview app.

 Note

For every operation in your Web service, a new AIF interface is generated.

3. Open the Message Monitoring Overview app.
4. Select the relevant AIF interface.

## Results

You've created an AIF interface for each operation in your asynchronous Web service. When you call an operation, you can monitor it in the Message Monitor Overview app.

## Related Information

[Set Up SOAP Communication \[page 1030\]](#)

### 4.2.9.3.8 Mail Communication

Create a mail destination and enable mail communication from SAP BTP, ABAP environment

[Sending Mails Using SMTP \[page 1032\]](#)

Send mails with the *Simple Message Transfer Protocol* (SMTP), using a mail server that is connected via the Cloud Connector.

#### 4.2.9.3.8.1 Sending Mails Using SMTP

Send mails using the Simple Message Transfer Protocol (SMTP).

You can send mails with the Simple Message Transfer Protocol (SMTP) using a mail server connected via the *SAP Business Technology Platform (SAP BTP)*, *Cloud Connector* or use a publicly available SMTP server.

For more information, see [Configure Access Control \(TCP\)](#)

## Application Programming Interface (API)

Use the `CL_BCS_MAIL_MESSAGE` class to create and send mails. You can specify sender and recipient. Furthermore, you can add textual body parts using class `CL_BCS_MAIL_TEXTPART` or binary attachments (e.g. PDF documents) represented by class `CL_BCS_MAIL_BINARYPART`, respectively.

With the send method of the API, the mail is sent via the mail server configured in the destination using the cloud connector. Sending is performed synchronously.

### Sample Code

```
try.
    data(lo_mail) = cl_bcs_mail_message->create_instance( ).
    lo_mail->set_sender( 'noreply@yourcompany.com' ).
    lo_mail->add_recipient( 'recipient1@yourcompany.com' ).
    lo_mail->add_recipient( iv_address = 'recipient2@yourcompany.com' iv_copy
= cl_bcs_mail_message->cc ).
    lo_mail->set_subject( 'Test Mail' ).
    lo_mail->set_main( cl_bcs_mail_txtpart->create_text_html( '<h1>Hello</
h1><p>This is a test mail.</p>' ) ).
    lo_mail->add_attachment( cl_bcs_mail_txtpart->create_text_plain(
        iv_content      = 'This is a text attachment'
        iv_filename     = 'Text_Attachment.txt'
    ) ).
    lo_mail->add_attachment( cl_bcs_mail_txtpart->create_instance(
        iv_content      = '<note><to>John</to><from>Jane</from><body>My nice
XML!</body></note>'
        iv_content_type = 'text/xml'
        iv_filename     = 'Text_Attachment.xml'
    ) ).

    data(lo_mail_status_monitor) = lo_mail->send( ).

    lo_mail_status_monitor->get_email_status(
        importing
            es_mail_status      = data(ls_mail_status)
            et_recipients_statuses = data(lt_recipients_statuses) ).

    catch cx_bcs_mail into data(lx_mail).
        "handle exceptions here
endtry.
```

## Bodyparts in a Message

Each mail message consists of one or more body parts (MIME parts). A body part consists of the text or binary content, the type and additional attributes like the filename for an attachment. A main body part (`SET_MAIN`) must be set and additional body parts can be set as attachments (`ADD_ATTACHMENT`) or alternatives (`ADD_MAIN_ALTERNATIVE`). Attachments are usually files that are attached to the mail. Alternative body parts are different representations of the main part of the email. For example, a mail can have an HTML representation and clients that can only display text or plain text will see this alternative body part.

The factory methods `CREATE_TEXT_PLAIN`, `CREATE_TEXT_HTML` of class `CL_BCS_MAIL_TEXTPART` can be used for the most common body parts. If a body part is to be sent with a different content-type, the `CREATE_INSTANCE` method can be used and a content-type can be passed.

## Sending Emails Asynchronously

In RESTful application programming (RAP) or in other transactional contexts, you may need to send an email asynchronously from time to time. This can be due to the performance, since external communication can take a while, or because you want to send an email when the COMMIT WORK statement is triggered at the end of a successful transaction.

An email can be sent asynchronously with the method `SEND_ASYNC`. See the following example:

### ↔ Sample Code

```
try.  
    data(lo_mail) = cl_bcs_mail_message->create_instance( ).  
    lo_mail->set_sender( 'noreply@yourcompany.com' ).  
    lo_mail->add_recipient( 'recipient1@yourcompany.com' ).  
    "...  
    lo_mail->send_async( ).  
catch cx_bcs_mail into data(lx_mail).  
    "handle exceptions here  
endtry.
```

After calling the `SEND_ASYNC` method, a background process is triggered. The process can be monitored in the Fiori app [Monitor Email Transmissions \[page 2727\]](#).

### ⚠ Caution

The process must call `COMMIT WORK` at the end of a transaction, otherwise the background process will not start and the email status will remain Waiting in the Monitor Email Transmissions app.

## Email Status Monitoring

The status of an email can be monitored using the Fiori app [Monitor Email Transmission](#). Additionally, it is possible to implement a monitoring instance directly in the backend for each email request. This monitoring instance, which is an implementation of the interface `IF_BCS_MAIL_STATUS_MONITOR`, provides information about the email status as well as the status of each recipient, including the SMTP response. The `send()` method provides the same statuses as exporting parameters.

To obtain this monitoring instance, you can either receive it as the return parameter of the methods `send()` or `send_async()`, or create a new instance using the factory method `create_mail_status_monitor()` of the class `CL_BCS_MAIL_MESSAGE`.

### 4.2.9.4 Developing APIs for Inbound Communication

Learn more about developing APIs for inbound communication.

You can create and expose services for external consumption to a communication partner. To create and expose services for external consumption, you create a communication scenario and assign the required services.

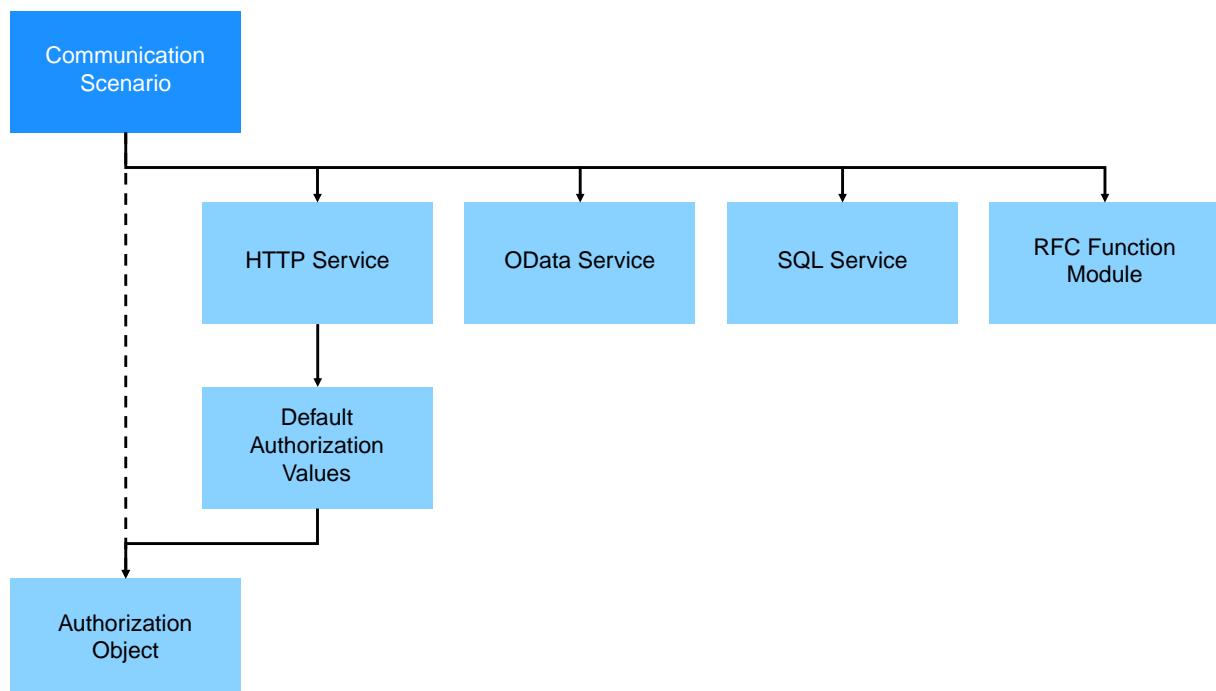
These inbound services can be:

- HTTP services, see [HTTP Service Development \[page 1038\]](#)
- Remote-enabled function modules, see [Develop a Remote-Enabled Function Module \(RFM\) \[page 1040\]](#)
- Services published via service bindings, such as OData or SQL services, see [Developing and Exposing an SQL Service in the ABAP System \[page 1208\]](#)

See also [Supported Protocols and Authentication Methods \[page 886\]](#).

Inbound services can be protected with an authorization object and protection against unauthorized activities. For each inbound service, authorization default values are defined that can later be used in the authorizations defined in the communication scenario. See [Granting Access Based on Activities for Communication Users \[page 852\]](#).

## ABAP System



## Related Information

[Communication Management \[page 883\]](#)

## 4.2.9.4.1 Create a Communication Arrangement for Inbound Communication with Service Key Type Basic

Learn how to quickly create a communication user and communication arrangement for an inbound communication scenario by using a service key of type basic.

### Prerequisites

- You have created a communication scenario.
- You have created a space. See [Create Spaces](#).
- You have the space developer role. See [Assigning the Space Developer Role to the Developer Users](#).

### Procedure

1. Log on to the cockpit and go to the subaccount that contains the space you'd like to navigate to. See [Navigate in the Cockpit](#).
2. In the navigation area, go to and choose your space.
3. Select from the navigation area and find your ABAP system service instance.
4. Click on and select [Create Service Key](#).
5. In the [New Service Key](#) dialog, enter a name for your service key and specify the parameters in JSON format as follows:

```
{  
  "scenario_id": "SAP_COM_XYZ",  
  "type": "basic"  
}
```

#### ⓘ Note

SAP\_COM\_XYZ is the ID of the communication scenario in your ABAP system.

basic is the type of service key that is needed to generate a communication user and communication arrangement for an inbound communication scenario.

6. **(Optional to step 5)** If your communication scenario has additional custom properties, you can automatically assign values to them by using the optional parameter additional\_properties.

```
{  
  "scenario_id": "SAP_COM_XYZ",  
  "type": "basic",  
  "additional_properties": [  
    {  
      "technical_name": "CUSTOM_PROPERTY_1",  
      "value": "VALUE_1"  
    },  
  ]  
}
```

```
        {
            "technical_name": "CUSTOM_PROPERTY_2",
            "value": "VALUE_2"
        }
    ]
}
```

#### ⓘ Note

CUSTOM\_PROPERTY\_1 and CUSTOM\_PROPERTY\_2 are the technical names of the properties as specified in your communication scenario. You can add any number of additional properties to the array `additional_properties` (CUSTOM\_PROPERTY\_3, CUSTOM\_PROPERTY\_4, and so on).

VALUE\_1, VALUE\_2, and so on are the values that should be assigned to the properties.

## Results

The communication user is generated and the credentials, depending on the selected type, are returned in the service key. The communication user receives authorizations for all services included in the communication scenario.

### 4.2.9.4.2 Configure Timeout of @sap/approuter Component of a Communication Arrangement for Inbound Communication with Service Key OAuth

By default, communication between the approuter and connected services is timed out after 30 seconds. This default value is suitable for most applications. However, for long-lasting data transfer, communication lasts longer. If you experience timeouts, you can set the service key parameter `abap_endpoint_timeout` to a value higher than 30 seconds.

## Prerequisites

- You have created a space. See [Create Spaces](#).
- You have the space developer role. See [Assigning the Space Developer Role to the Developer Users](#).

## Procedure

1. Log on to the cockpit and go to the subaccount that contains the space you'd like to navigate to. See [Navigate in the Cockpit](#).
2. In the navigation area, go to  [Cloud Foundry](#)  [Spaces](#) and choose your space.

3. Select from the navigation area and navigate to your ABAP system service instance.
4. Click on and select *Create Service Key*.
5. In the *New Service Key* dialog, enter a name for your service key and specify the parameter in JSON format as follows:

```
{  
    "abap_endpoint_timeout": 600000  
}
```

#### Note

You can use s (seconds) or m (minutes) to define the value.

If no unit is defined, the value is interpreted as seconds.

The maximum value is 600s or 10m.

## Related Information

[Integration with Business Services](#)

### 4.2.9.4.3 HTTP Inbound Communication

Learn how to set up and enable HTTP inbound communication.

[HTTP Service Development \[page 1038\]](#)

#### 4.2.9.4.3.1 HTTP Service Development

With the ABAP environment, you can develop HTTP services in your ABAP Development Tools (ADT) in Eclipse.

For the implementation of your HTTP service, we provide the interface `IF_HTTP_SERVICE_EXTENSION` with HTTP request/response parameters, giving you the full flexibility to build an HTTP service of your choice. See [Working with the HTTP Service Editor](#) and [Tutorial: Create an HTTP Service](#).

## Related Information

[ABAP Development User Guide](#)

[Protection Against CSRF Attacks \[page 1039\]](#)

## 4.2.9.4.3.1.1 Protection Against CSRF Attacks

Protect against cross-side request forgery (CSRF) attacks by enabling CSRF token handling in the HTTP service.

To protect an HTTP service against CSRF attacks, you must implement the method `HANDLE_REQUEST` in the interface `IF_HTTP_SERVICE_EXTENSION` of the service. If this method is implemented, within a session each caller of the service must :

- Request a CSRF token by setting the header `X-CSRF-Token` with value `fetch`.
- Provide the specific CSRF token with each request of this session in header `X-CSRF-Token`, which is validated by the server.

The method `HANDLE_REQUEST`:

- Provides a CSRF token and adds it to the HTTP response, if the value `fetch` is present in the corresponding header field.
- Validates the CSRF token, if it is present in the header field of the request.
- Returns an error with the HTTP response, if validation fails.

### ⚠ Caution

If you have implemented the method `IF_HTTP_SERVICE_EXTENSION` for an HTTP service, all calls that do not request a CSRF token and provide it in the respective session are rejected. Only clients enabled for CSRF token handling can perform successful calls to the service in this case.

### Example

The code below shows an example for implementing the method `HANDLE_REQUEST` of the interface `IF_HTTP_SERVICE_EXTENSION` to enable correct CSRF token handling:

#### ↔ Sample Code

```
METHOD if_http_service_extension~handle_request.  
  DATA(lv_valid) = cl_http_service_utility=>handle_csrf(  
    EXPORTING  
      request = request  
      response = response  
    ).  
  CHECK lv_valid = abap_true.  
  //Start the implementation of the HTTP service  
  //Start the application logic  
ENDMETHOD.
```

## 4.2.9.4 RFC Inbound Communication

Learn how to set up and enable RFC inbound communication.

#### [Develop a Remote-Enabled Function Module \(RFM\) \[page 1040\]](#)

Develop RFMs in ABAP Development Tools (ADT).

#### [Set Up the Cloud Connector for Inbound RFC from On-Premise Systems \(Deprecated\) \[page 1041\]](#)

Configure inbound connectivity to the ABAP environment to consume a remote-enabled function module (RFM) from an on-premise ABAP system through the Cloud Connector.

#### [Set Up Inbound RFC from On-Premise Systems Using WebSocket RFC \[page 1043\]](#)

Learn how to set up an inbound RFC connection from an on-premise system to the ABAP environment using WebSocket RFC. Get instructions on creating a communication scenario and arrangement, as well as the necessary settings in the on-premise system.

### **4.2.9.4.4.1 Develop a Remote-Enabled Function Module (RFM)**

Develop RFMs in ABAP Development Tools (ADT).

You can develop RFMs in ABAP Development Tools (ADT) for Eclipse. For more information, see [Creating an ABAP Function Module](#).

In this tool, you create the respective RFM as you are used to in an on-premise system. Objects that are required to consume the RFM remotely are created automatically, such as authorization default values and the respective inbound service.

#### ⓘ Note

To enable automatic object creation for an RFM in ADT, choose *Properties* and select RFC as *<Processing Type>*. The default processing type is *Normal*. When activating an RFM of type *Normal*, no objects are generated.

For detailed information on consuming an RFM in an inbound communication scenario, see [Set Up the Cloud Connector for Inbound RFC from On-Premise Systems \(Deprecated\) \[page 1041\]](#).

### **RFC Metadata Integration (Communication Scenario SAP\_COM\_0636)**

Instead of using classic RFC via CPI-C, you can also use RFC via WebSocket with SAP Java Connector (JCo) or other RFC connectors.

Optionally, you can configure these external RFC connectors to use a dedicated user to retrieve the metadata.

#### ⓘ Note

This scenario also applies for calls routed through the Cloud Connector.

Communication scenario SAP\_COM\_0636 lets you separate metadata access for RFC calls by RFC client libraries from the actual RFC application call.

It provides access to the following function modules:

- [RFC\\_METADATA\\_GET](#)
- [RFC\\_FUNCTION\\_SEARCH](#)
- [RFC\\_GET\\_FUNCTION\\_INTERFACE](#)
- [DDIF\\_FIELDINFO\\_GET](#)

**ⓘ Note**

When using WebSocket RFC, you must use a host name that follows the pattern  
<myTenantHost>.abap.hana.ondemand.com.

#### 4.2.9.4.4.2 Set Up the Cloud Connector for Inbound RFC from On-Premise Systems (Deprecated)

Configure inbound connectivity to the ABAP environment to consume a remote-enabled function module (RFM) from an on-premise ABAP system through the Cloud Connector.

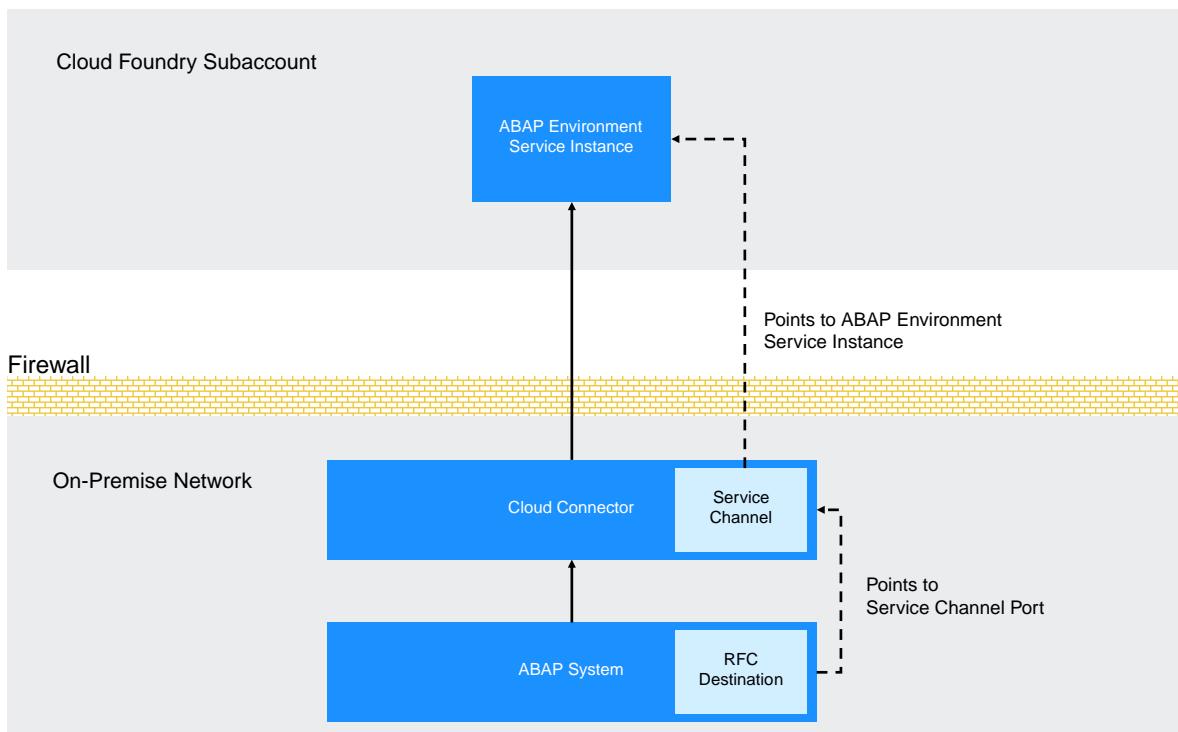
#### Context

RFC connections from on-premise systems to the ABAP environment can be performed through a so called Cloud Connector service channel.

**ⓘ Note**

Using Cloud Connector service channels is deprecated. Use WebSocket RFC instead. HTTP(S) or WebSocket RFC connections do not require the Cloud Connector because they can be opened directly on the Internet.

To establish an RFC connection from an on-premise system to the ABAP environment, the Cloud Connector opens an RFC port just like an RFC gateway and acts as a local proxy for the cloud system. Connections to this port are routed through a secure tunnel to SAP BTP.



In the Cloud Connector, you create a service channel with a local instance number. In the on-premise ABAP system, you correspondingly create an RFC destination in transaction SM59 with the hostname of the Cloud Connector in your on-premise network and the same instance number.

### ① Note

If you are using different ABAP instances on SAP BTP, create a service channel for each instance, choose an arbitrary instance number in the service channel settings of the Cloud Connector configuration, and use the same instance number in the RFC destination of your on-premise system (transaction SM59) to identify the called ABAP instance. Since the instance number is defined as a two-digit number, the number of target systems you can configure in the Cloud Connector is limited to 99.

## Prerequisites

- You have developed a communication scenario that contains the RFM to be called.
- You have created a communication arrangement of that scenario with a communication system and communication user.

### ① Note

To define authorizations for the communication user, use the authorization default values on inbound service level. These authorizations are automatically assigned to the respective communication user.

## Restrictions

- Only remote function modules that are part of a communication scenario can be called from an on-premise system. Most others are blocked, except for the system function module RFCPING.
- An inbound RFC connection supports only *basic authentication* as authentication type.
- The user used for authentication must be a *communication user*, a *business user* is not allowed.
- The user name you can enter in an RFC destination of an on-premise ABAP system is limited to 12 characters.

## Procedure

1. To establish a service channel, configure it in the Cloud Connector as described in [Configure a Service Channel for RFC](#).

### ⓘ Note

When creating a service channel for the ABAP environment, the URL of the ABAP instance (tenant host) follows the pattern `<myTenantHost>.abap.hana.ondemand.com` (not `<myTenantHost>-api...`, which is valid only for connections to S/4HANA Cloud). Make sure you create the service channel for the correct subaccount.

2. Create an RFC destination in the on-premise system:

- Use connection type 3 (RFC connection to an ABAP system).
- **Do not use load balancing.** As target host, enter the hostname of Cloud Connector. As instance number, enter the local instance number you have used in the configuration of the service channel in the Cloud Connector.
- Provide logon credentials for a communication user, including the client information. The ABAP environment always uses client 100.
- Use the connection test in transaction SM59 to verify that the service channel works correctly.

### ⓘ Note

When calling an RFM, you must write its name in *upper case*.

## 4.2.9.4.4.3 Set Up Inbound RFC from On-Premise Systems Using WebSocket RFC

Learn how to set up an inbound RFC connection from an on-premise system to the ABAP environment using WebSocket RFC. Get instructions on creating a communication scenario and arrangement, as well as the necessary settings in the on-premise system.

## Context

Use WebSocket RFC to establish an RFC connection from an on-premise system to the ABAP environment. WebSocket RFC makes it possible to call RFC function modules of the ABAP environment using the API URL of the cloud system using TLS-encrypted WebSocket connections.

### ⓘ Note

All WebSocket-RFC-connections are TLS-encrypted. WebSocket RFC always uses HTTPS.

## Procedure

1. Create a Communication Scenario that defines all RFMs to be called on the server side (inbound scenario). See [Communication Scenario \[page 883\]](#) for more information.
2. Create a Communication Arrangement with a Communication System and a Communication User. See the following for more information:
  - [Communication Arrangement \[page 885\]](#)
  - [Communication System \[page 884\]](#)
  - [Communication User \[page 884\]](#)

## Settings in the On-Premise System

In the on-premise system, the following parameters must be set in transaction SM59.

Create a destination of connection type w:

### *Technical Settings*

Host	The API-URL defined in the Communication Arrangement without the <b>https://</b> prefix
------	---

Port	<b>443</b>
------	------------

### *Logon & Security*

Explicit Client	by Hostname
Authentication Method	by User/Password

### ⓘ Note

If you choose user/password authentication, activate **Alias User**.

by X.509 Certificate

### Note

These settings are available with SAP S/4HANA 1909. For older releases, you need to transform from a CPIC-based RFC to WebSocket RFC via Business Connector. See [SAP Business Connector](#) for more information.

## Related Information

[Develop a Remote-Enabled Function Module \(RFM\) \[page 1040\]](#)

### 4.2.9.4.5 SOAP Inbound Communication

Learn how to set up and enable SOAP inbound communication.

#### 4.2.9.4.5.1 Generate Provider Proxies From WSDL Files

SOAP-based provider proxies are generated using a so-called SOAP Provider Model (SPRV).

### Prerequisites

You have created a communication scenario as described in [Defining a Communication Scenario Including Authorization Values \[page 858\]](#).

### Context

With the SOAP Provider Model (SPRV), you can generate the proxy objects that are required for providing SOAP services, based on a WSDL file that describe your service. The SPRV is used to provide a corresponding inbound service for an exposed outbound service.

### Example

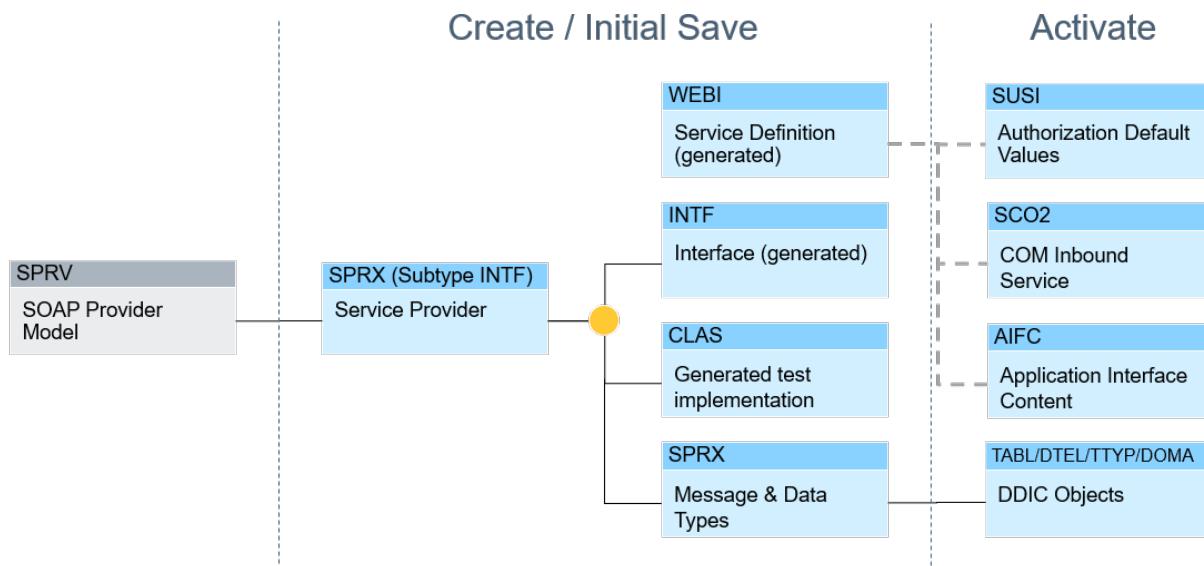
An external system triggers a confirmation in your system to inform the external system about the creation of an appointment activity. To receive this information, a provider proxy for this service must be implemented in your system. This provider proxy is generated by the SPRV.

### **⚠ Restriction**

In ABAP development tools for Eclipse, it is not possible to model SOAP web services. Therefore, the SPRV is not intended to provide a SOAP service.

During and after activation, the following objects are generated:

- Proxy objects
- DDIC-objects, e.g. the required data elements
- The service definition
- Application Interface Framework objects for monitoring
- The required inbound services for the communication scenario
- The authorization default values



### **⚠ Restriction**

It is not possible to model web services in ABAP development tools for Eclipse. To use the SPRV, you must use a WSDL file that describes the service interface.

## **Procedure**

1. Based on the WSDL file of the service you want to provide, create an SPRV as described in [Creating SOAP Service Providers](#).
2. Activate the SPRV.
3. Add the generated inbound service to the relevant communication scenario.
4. Publish the communication scenario locally to make it available for the administrator in SAP Fiori launchpad.

## Next Steps

The SPRV generates an implementing class of the service you want to provide. To provide the functionality of your service, add the required application logic to the generated implementing class. Afterwards, activate the implementing class.

### Note

When deleting the SPRV, the implementing class is not deleted to avoid data loss. All other generated objects that depend on the SPRV are deleted with the SPRV.

Based on the communication scenario, the administrator can create a communication system, inbound user, and communication arrangement, to make the service available for consumption. For more information, see [Communication Management \[page 883\]](#), and [SOAP Outbound Communication \[page 1014\]](#)

### 4.2.9.5 SAP-Provided APIs

[Inbound Service: Business User \[page 1048\]](#)

[Inbound Service: Business User - Read \[page 1065\]](#)

[Business User - Read Metadata \[page 1080\]](#)

[Integrating Business Role Change Documents \[page 1087\]](#)

You can use this scenario to read the change documents of business roles.

[Integrating Business User Change Documents \[page 1090\]](#)

You can use this scenario to read the change documents of business users.

[Integrating Business Role Data \(Read\) \[page 1092\]](#)

You can use this communication scenario to read business role data.

[Integrating Business User Logon Details \(Read\) \[page 1099\]](#)

You can use this communication scenario to read business user logon details.

[Integrating Support User Data \(Read\) \[page 1104\]](#)

You can use this communication scenario to read support user data.

[Integrating Communication Arrangement Data \(Read\) \[page 1107\]](#)

You can use this communication scenario to read communication arrangement data.

[Integrating Communication Management Data \(Read and Write\) \[page 1114\]](#)

You can use this communication scenario to read and write communication arrangement, communication system and communication user data.

[Integrating Communication System Data \(Read\) \[page 1119\]](#)

You can use this communication scenario to read communication system data.

[Integrating Communication User Data \(Read\) \[page 1126\]](#)

You can use this communication scenario to read communication user data.

[Integrating Content Security Policy Data \(Read\) \[page 1131\]](#)

You can use this communication scenario to read Content Security Policy (CSP) data.

[Integrating Protection Allowlist Data \(Read\) \[page 1134\]](#)

You can use this communication scenario to read protection allowlist data.

[Integrating Certificate Data \(Read\) \[page 1141\]](#)

[Integrating Cross-domain Identity Management \[page 1151\]](#)

You can use this communication scenario to maintain data via SCIM.

[Manage Git Repository \[page 1156\]](#)

The Manage Git Repository API allows you to manage software components (Git repositories) on an SAP BTP ABAP Environment system.

### 4.2.9.5.1 Inbound Service: Business User

Technical name: MANAGEBUSINESSUSERIN

This synchronous inbound SOAP service enables you to create, update, and delete business users from your external data sources, such as an identity management system. Deleting business users doesn't mean you've actually deleted them yet. The user assigned to the business user is deleted and the `MarkedForArchivingIndicator` has been set. Only during the ILM process the business user is physically deleted.

You can assign business role IDs to the users at the node `Role`.

We recommend processing blocks of 10 users to a maximum of 100 users. Otherwise, the target system may time out.

This service supports the business users Employee (BUP003).

#### ⚠ Caution

This service directly influences the data and authorizations of business users. Changes are effective immediately in the target system.

Make sure to maintain only those authorizations that are intended for what a user needs to do in the system. Not doing so can cause security issues.

### Service Request

The service is structured into the following two top-level nodes:

**Message Header (MessageHeader)**

The service message header is not in use in this service.

**Business User (BusinessUser)**

The service nodes contain the service's business data.

#### ⓘ Note

In the following table, attributes are marked in blue.

Nodes and Fields for the BusinessUser Node

Node or Field	Description	Maximum Field		
		Length	Cardinality	Necessity
PersonExternalID	The Person External ID is the ID of the employee.	60	0..1	Mandatory for create for business partner category (BUP003 - Employee).
				Optional for update (at least one of the person IDs is mandatory.)
PersonID	Person ID is the ID of the business partner.	10	0..1	Optional (At least one of the person IDs is mandatory.)
PersonUUID	Person UUID is the unique ID of the business partner.	36	0..1	Optional (At least one of the person IDs is mandatory.)
BusinessPartnerRoleCode	Business Partner Role Code	6	0..1	Mandatory for create for Business partner category (BUP003 - Employee).
				Optional for update.

Node or Field	Description	Maximum Field Length	Cardinality	Necessity
MarkedForArchivingIndicator	<p>Mark for Archiving</p> <p>Set to <b>True</b>:</p> <ul style="list-style-type: none"> <li>The business user will be archived</li> <li>The actionCode [1] for User must be set to 02</li> </ul> <p>Set to <b>False</b>:</p> <ul style="list-style-type: none"> <li>The business user will be reactivated (Undo Archive)</li> <li>The actionCode [1] for User must be set to 02</li> </ul>	0..1	Optional	
ValidityPeriod	<p>Format: YYYY-MM-DD</p> <p>Cardinality: 0..1</p> <p>By default, the system date is set.</p>	0..1	Optional	
EndDate	<p>Format: YYYY-MM-DD</p> <p>By default, 9999-12-31 is set.</p>	0..1	Optional	
PersonalInfoFormOfAddressInformation	Form of address	4	0..1	Optional

Node or Field	Description	Maximum Field		
		Length	Cardinality	Necessity
Cardinality: 0..1				
FirstName	First name	40	0..1	Optional
LastName	Last name	40	0..1	Mandatory
	This field is mandatory.			
PersonFullName	Person full name	80	0..1	Optional
AcademicTitle	Academic title	4	0..1	Optional
CorrespondenceLanguage	Correspondence language	9	0..1	Optional
MiddleName	Middle name	40	0..1	Optional
AdditionalLastName	Additional last name	40	0..1	Optional
BirthName	Birth name	40	0..1	Optional
NickName	Nick name	40	0..1	Optional
Initials	Initials	10	0..1	Optional
AcademicSecondTitle	Academic second title	4	0..1	Optional
LastNamePrefix	Last name prefix	4	0..1	Optional
LastNameSecondPrefix	Last name second prefix	4	0..1	Optional
NameSupplement	Name supplement	4	0..1	Optional

Node or Field		Description	Maximum Field		
			Length	Cardinality	Necessity
	actionCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 02 - Update</li> <li>• 03 - Delete</li> </ul> <p>Mandatory if [2] is not set and personal information data are given.</p>	2	optional	Optional
User (only for Cloud)	UserName	User name/ Alias	40	0..1	Optional
Cardinality: 0..1	LogonLanguageCode	Logon language	9	0..1	Optional
<b> ⓘ Note</b> For more information about default values, see <a href="#">Defaulting of User Attributes</a> .					

Node or Field	Description	Maximum Field Length	Cardinality	Necessity
DateFormatCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 1 - DD.MM.YY YY (Gregorian Date)</li> <li>• 2 - MM/DD/YYYY (Gregorian Date)</li> <li>• 3 - MM-DD-YYYY (Gregorian Date)</li> <li>• 4 - YYYY.MM.DD (Gregorian Date)</li> <li>• 5 - YYYY/MM/DD (Gregorian Date)</li> <li>• 6 - YYYY-MM-DD (Gregorian Date, ISO 8601)</li> <li>• 7 - GYY.MM.DD (Japanese Date)</li> <li>• 8 - GYY/MM/DD (Japanese Date)</li> <li>• 9 - GYY-MM-DD (Japanese Date)</li> </ul>	2	0..1	Optional

Node or Field	Description	Maximum Field		
		Length	Cardinality	Necessity
	<ul style="list-style-type: none"> <li>• A - YYYY/MM/ DD (Islamic Date 1)</li> <li>• B - YYYY/MM/ DD (Islamic Date 2)</li> <li>• C - YYYY/MM/ DD (Iranian Date)</li> </ul>			
DecimalFormatCode	You can use the following values: <ul style="list-style-type: none"> <li>• 1.234.567,89</li> <li>• X - 1,234,567,89</li> <li>• Y - 1234 567,89</li> </ul>	2	0..1	Optional
TimeZoneCode	Time zone	10	0..1	Optional

Node or Field	Description	Length	Cardinality	Maximum Field Necessity
TimeFormatCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 0 - 24 Hour Format (Example: 12:05:10)</li> <li>• 1 - 12 Hour Format (Example: 12:05:10 PM)</li> <li>• 2 - 12 Hour Format (Example: 12:05:10 pm)</li> <li>• 3 - Hours from 0 to 11 (Example: 00:05:10 PM)</li> <li>• 4 - Hours from 0 to 11 (Example: 00:05:10 pm)</li> </ul>	2	0..1	Optional
LockedIndicator	Locked indicator	5	0..1	Optional
ValidityPeriod	<p>Format: YYYY-MM-DD</p> <p>Cardinality: 1</p> <p>If the start date is not maintained for the User, the StartDate for the BusinessUser is entered.</p>		1	Optional

Node or Field	Description	Length	Cardinality	Maximum Field Necessity
EndDate	Format:  YYYY-MM-DD  If the <code>EndDate</code> is not maintained, it is set to 9999-12-31.		1	Optional
Role  Cardinality: 0..unbounded	RoleName  <code>actionCode</code>  You can use the following values: <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 03 - Delete</li> </ul> Mandatory if [6] is not set and role name data is given.	40	1	Optional
GlobalUserID	Used to uniquely identify your business user across different systems.	36		Optional
UserGroupCode	Assign a business user to an existing user group.	12	0..1	Optional

Node or Field		Description	Length	Maximum Field Cardinality	Necessity
	actionCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 02 - Update</li> <li>• 03 - Delete</li> </ul> <p>Mandatory if [3] is not set and user data (UserName and Role) are given.</p>	2	Optional	Optional
	[6] roleListCompleteTransmissionIndicator	<p>CTI for the Role node</p> <p>See <a href="#">CTI note</a> [page 1061] for more information.</p>		Optional	Optional
UserAssignment (only for on-premise)	User ID	User ID	12	1	N/A
Cardinality: 0..1	actionCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 02 - Update</li> <li>• 03 - Delete</li> </ul> <p>Mandatory if [4] is not set and User ID data are given.</p>	2	Optional	N/A
WorkplaceInformation	EmailAddress	Email address	241	0..1	N/A
Cardinality: 0..2	PhoneInformation	<p>Phone type</p> <ul style="list-style-type: none"> <li>• B - Business</li> <li>• C - Cell</li> </ul>	1	1	N/A
One set of phone information	PhoneType				

Node or Field	Description	Maximum Field		
		Length	Cardinality	Necessity
CountryDialingCode	Country dialing code  Used for both phone types.	10	0..1	N/A
PhoneNumberAreaID	Phone number area code  Used only for phone type B.	10	0..1	N/A
PhoneNumbersSubscriberID	Phone number subscriber ID  Used for both phone types.	30	0..1	N/A
PhoneNumberExtension	Phone number extension  Used only for phone type B.	10	0..1	N/A
actionCode	You can use the following values: <ul style="list-style-type: none"><li>• 01 - Create</li><li>• 02 - Update</li><li>• 03 - Delete</li></ul> Mandatory if [7] is not set and phone data is given.	2	Optional	N/A
FunctionalTitleName	Functional title name	40	0..1	N/A
Department	Department name	40	0..1	N/A
RoomNumber	Room number	10	0..1	N/A
Building	Building name	10	0..1	N/A

Node or Field		Description	Length	Maximum Field Cardinality	Necessity
	actionCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 02 - Update</li> <li>• 03 - Delete</li> </ul> <p>Mandatory if [5] is not set and workplace information data is given.</p>	2	Optional	N/A
	[7] phoneInformationListCompleteTransmissionIndicator	<p>CTI for the PhoneInformation node</p> <p>See <a href="#">CTI note [page 1061]</a> for more information.</p>		Optional	N/A
Relationship (only for on-premise)	Partner1	<p>BusinessPart nerID</p> <hr/> <p>BusinessPart nerUUID</p> <hr/> <p>BusinessPart nerExternalID D</p> <hr/> <p>BusinessPart nerExternalID DCategoryCode</p>		N/A	
	Partner2	<p>BusinessPart nerID</p> <hr/> <p>BusinessPart nerUUID</p> <hr/> <p>BusinessPart nerExternalID D</p>	0..1	N/A	N/A

Node or Field	Description	Length	Cardinality	Maximum Field Necessity
BusinessPart nerExternalI DCategoryCod e			0..1	N/A
RelationshipCategoryCode			0..1	N/A
actionCode			Optional	N/A
[1] actionCode	<p>You can use the following values:</p> <ul style="list-style-type: none"> <li>• 01 - Create</li> <li>• 02 - Update</li> <li>• 03 - Delete</li> </ul> <p>This attribute is mandatory.</p>		Optional	Optional
[2] personalInformationListCompleteTransmissionIndicator	<p>CTI for the PersonalInfo rimation node</p> <p>See <a href="#">CTI note [page 1061]</a> for more information.</p>		Optional	Optional
[3] userListCompleteTransmissionIndicator	<p>CTI for the User node</p> <p>See <a href="#">CTI note [page 1061]</a> for more information.</p>		Optional	Optional
[4] userAssignmentListCompleteTransmissionIndicator	<p>CTI for the UserAssignme nt node</p> <p>See <a href="#">CTI note [page 1061]</a> for more information.</p>		Optional	Optional

Node or Field	Description	Length	Maximum Field Cardinality	Necessity
[5] workplaceInformationListCompleteTransmissionIndicator	CTI for the WorkplaceInformation node  See <a href="#">CTI note [page 1061]</a> for more information.		Optional	Optional
[5] relationshipListCompleteTransmissionIndicator			Optional	Optional

## ⓘ Note

### Complete Transmission Indicator (CTI)

The complete transmission indicator represents the state of the data in the inbound XML. There are multiple transmission indicators that represent the state of the corresponding segment. The transmission indicator is interpreted by the web service:

- If a complete transmission indicator is set to true, the existing data is overwritten by the newly entered data.
- If a complete transmission indicator is set to false, the value of the action code of the corresponding node applies.
- If an action code and the complete transmission indicator are set, then the action code of the corresponding node applies.

## Sample Payload

### ↔ Sample Code

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <xmlns:aba="http://sap.com/xi/ABA">
    <soapenv:Header/>
    <soapenv:Body>
      <aba:BusinessUserBundleMaintainRequest_sync>
        <!--1 or more repetitions:-->
        <BusinessUser actionCode="01"
personalInformationListCompleteTransmissionIndicator="false"
userListCompleteTransmissionIndicator="false"
userAssignmentListCompleteTransmissionIndicator="false"
workplaceInformationListCompleteTransmissionIndicator="false">
          <PersonExternalID>Muster01</PersonExternalID>
          <BusinessPartnerRoleCode>BUP003</BusinessPartnerRoleCode>
          <PersonalInformation actionCode="01">
            <FormOfAddress>0002</FormOfAddress>
            <FirstName>Max</FirstName>
            <LastName>Muster</LastName>
            <PersonFullName>Prof. Dr. Max Muster</PersonFullName>
            <AcademicTitle>0002</AcademicTitle>

          <CorrespondenceLanguage>D</CorrespondenceLanguage>
        </BusinessUser>
      </aba:BusinessUserBundleMaintainRequest_sync>
    </soapenv:Body>
  </soapenv:Envelope>

```

```

<MiddleName>Michael</MiddleName>
<AcademicSecondTitle>0001</AcademicSecondTitle>
<BirthName>Milli</BirthName>
<NickName>Maxi</NickName>
<LastNamePrefix>0001</LastNamePrefix>
</PersonalInformation>
<User actionCode="01"
roleListCompleteTransmissionIndicator="false">
    <!--Optional:>
    <UserName>MAXMUSTER01</UserName>
    <LogonLanguageCode>DE</LogonLanguageCode>
    <LockedIndicator>false</LockedIndicator>
    <Role actionCode="01">
        <RoleName>SAP_BR_MANAGER</RoleName>
    </Role>
    <Role actionCode="01">
        <RoleName>SAP_BR_BPC_EXPERT</RoleName>
    </Role>
</User>
<WorkplaceInformation actionCode="01"
phoneInformationListCompleteTransmissionIndicator="true">
    <EmailAddress>Max.Muster01@Test.com</EmailAddress>
    <PhoneInformation actionCode="01">
        <PhoneType>C</PhoneType>
        <CountryDialingCode>+49</CountryDialingCode>
        <PhoneNumberSubscriberID>0160123456</
PhoneNumberSubscriberID>
        </PhoneInformation>
        <PhoneInformation actionCode="01">
            <PhoneType>B</PhoneType>
            <CountryDialingCode>+49</CountryDialingCode>
            <PhoneNumberAreaID>06227</PhoneNumberAreaID>
            <PhoneNumberSubscriberID>7</PhoneNumberSubscriberID>
            <PhoneNumberExtension>12345</PhoneNumberExtension>
        </PhoneInformation>
        <FunctionalTitleName>TESTER</FunctionalTitleName>
        <Department>QUALITY</Department>
        <RoomNumber>C1.23</RoomNumber>
        <Building>WDF01</Building>
    </WorkplaceInformation>
</BusinessUser>
<BusinessUser actionCode="01"
personalInformationListCompleteTransmissionIndicator="false"
userListCompleteTransmissionIndicator="false"
userAssignmentListCompleteTransmissionIndicator="false"
workplaceInformationListCompleteTransmissionIndicator="false">
    <PersonExternalID>MINIMUSTER01</PersonExternalID>
    <BusinessPartnerRoleCode>BUP003</BusinessPartnerRoleCode>
    <PersonalInformation actionCode="01">
        <FormOfAddress>0001</FormOfAddress>
        <FirstName>Mini</FirstName>
        <LastName>Muster</LastName>
        <PersonFullName>Prof. Dr. Mini Muster</PersonFullName>
        <AcademicTitle>0002</AcademicTitle>
        <CorrespondenceLanguage>D</CorrespondenceLanguage>
        <AcademicSecondTitle>0001</AcademicSecondTitle>
        <LastNamePrefix>0001</LastNamePrefix>
    </PersonalInformation>
    <User actionCode="01"
roleListCompleteTransmissionIndicator="false">
        <!--Optional:>
        <UserName>MINIMUSTER01</UserName>
        <LogonLanguageCode>DE</LogonLanguageCode>
        <LockedIndicator>false</LockedIndicator>
        <Role actionCode="01">
            <RoleName>SAP_BR_MANAGER</RoleName>
        </Role>
        <Role actionCode="01">

```

```

        <RoleName>SAP_BR_BPC_EXPERT</RoleName>
    </Role>
</User>
<WorkplaceInformation actionCode="01"
phoneInformationListCompleteTransmissionIndicator="true">
    <EmailAddress>Mini.Muster01@Test.com</EmailAddress>
    <PhoneInformation actionCode="01">
        <PhoneType>C</PhoneType>
        <CountryDialingCode>+49</CountryDialingCode>
        <PhoneNumberSubscriberID>0160123456</
    PhoneNumberSubscriberID>
    </PhoneInformation>
    <PhoneInformation actionCode="01">
        <PhoneType>B</PhoneType>
        <CountryDialingCode>+49</CountryDialingCode>
        <PhoneNumberAreaID>06227</PhoneNumberAreaIDs>
        <PhoneNumberSubscriberID>7</PhoneNumberSubscriberID>
        <PhoneNumberExtension>12345</PhoneNumberExtension>
    </PhoneInformation>
    <FunctionalTitleName>TESTER</FunctionalTitleName>
    <Department>QUALITY</Department>
    <RoomNumber>C1.23</RoomNumber>
    <Building>WDF01</Building>
</WorkplaceInformation>
</BusinessUser>
</aba:BusinessUserBundleMaintainRequest_sync>
</soapenv:Body>
</soapenv:Envelope>

```

## Service Response

You receive a confirmation message response for each bundle of business users you send. If the service request is processed, a confirmation message is sent. This contain crucial information provided by the fields PersonExternalID, PersonID, and PersonUUID for each business user of the bundle.

The following table provides an overview of the response structure for the BusinessUser service node.

Field or Node		Maximum Field		
		Description	Length	Cardinality
PersonExternalID		Person External ID	60	0..1
PersonID		Person ID	10	0..1
PersonUUID		Person UUID	36	0..1
Log	BusinessDocumentProcessingResultCode	Not in use	2	0..1
Cardinality: 1				

Field or Node	Description	Maximum Field Length	Cardinality
MaximumLogItemSeverityCode	If several messages are stored for a business user, the maximum of all received severity codes the most severe level will be shown.	1	0..1
Item	TypeID	Message number	40
Cardinality: 0..unbounded	CategoryCode	Not in use	15
	SeverityCode	Severity code definition:	1
		<ul style="list-style-type: none"> <li>• 1 - Information</li> <li>• 2 - Warning</li> <li>• 3 - Error</li> </ul>	0..1
Note	Contains the message texts.	200	1
WebURI	Not in use		0..1

## Error Codes

Error Code	Description
104	<p>Combination of Ext. ID &amp;1 and ID &amp;2 inconsistent. Processing cancelled.</p> <p>PersonExternalID and PersonID have a 1:1 relationship. Enter thePersonID that corresponds with the PersonExternalID.</p>
105	<p>Combination of Ext. ID &amp;1 and UUID &amp;2 inconsistent.</p> <p>Person ExternalID and PersonUUID have a 1:1 relationship.</p> <p>Enter thePersonUUID that corresponds with the PersonExternalID</p>

## Constraints

This service does not support:

- Freelancer (BBP010) business users

## Additional Information

### Note

For more details about Communication Management, see [Communication Management \[page 2591\]](#).

## 4.2.9.5.2 Inbound Service: Business User - Read

Technical name: QUERYBUSINESSUSERIN

This synchronous inbound SOAP service enables you to read users from your external data source such as an identity management system in SAP S/4HANA Cloud.

## Service Request

The service is structured into the following two top-level nodes:

### **Business User (BusinessUser)**

The service node contains the search parameters.

Nodes and Fields for the BusinessUser Node

Field or Node	Description	Maximum Field Length	Cardinality
PersonExternalIDIntervalBoundaryTypeCode	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryPersonExtID is set.</p>	1	1
LowerBoundaryPersonExtID	Employee name	60	0..1
UpperBoundaryPersonExtID		60	0..1

Field or Node		Description	Maximum Field Length	Cardinality
PersonIDInterval Cardinality: 0..un- bounded	IntervalBoundaryT ypeCode	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryPers onID is set.</p>	1	1
	LowerBoundaryPers onID		10	0..1
	UpperBoundaryPers onID		10	0..1

Field or Node	Description	Maximum Field Length	Cardinality
BusinessPartnerRoleCodeInterval	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryBusinessPartnerRoleCode is set.</p>	1	1
LowerBoundaryBusinessPartnerRoleCode	Only business partner role code BUP003 (Employee) is supported.	6	0..1
MarkedForArchivingIndicator	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• True</li> <li>• False</li> </ul>	1	0..1
LowerBoundaryMarkedForArchivingIndicator		1	0..1

Field or Node	Description	Maximum Field Length	Cardinality
UserIDInterval Cardinality: 0..un- bounded	<p>IntervalBoundaryT ypeCode</p> <p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryUser ID is set.</p>	1	1
LowerBoundaryUser ID		12	0..1
UpperBoundaryUser ID		12	0..1

Field or Node		Description	Maximum Field Length	Cardinality
UserNameInterval Cardinality: 0..un- bounded	IntervalBoundaryT ypeCode	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryUser Name is set.</p>	1	1
LowerBoundaryUser Name			40	0..1
UpperBoundaryUser Name			40	0..1

Field or Node	Description	Maximum Field Length	Cardinality
FirstNameInterval Cardinality: 0..un- bounded	<p>IntervalBoundaryT ypeCode</p> <p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1- Equal No upper boundary value must be set.</li> <li>• 3 - Between Upper boundary value is mandatory.</li> <li>• 6 - Lower than Upper boundary value is optional.</li> <li>• 7 - Lower equal Upper boundary value is optional.</li> <li>• 8 - Greater than Upper boundary value is optional.</li> <li>• 9 - Greater equal Upper boundary value is optional.</li> </ul> <p>This field is mandatory if LowerBoundaryFirs tName is set.</p>	1	1
LowerBoundaryFirs tName		35	0..1
UpperBoundaryFirs tName		35	0..1

Field or Node		Description	Maximum Field Length	Cardinality
LastNameInterval Cardinality: 0..un- bounded	IntervalBoundaryT yTypeCode	The following values exist: <ul style="list-style-type: none"><li>• 1- Equal No upper boundary value must be set.</li><li>• 3 - Between Upper boundary value is mandatory.</li><li>• 6 - Lower than Upper boundary value is optional.</li><li>• 7 - Lower equal Upper boundary value is optional.</li><li>• 8 - Greater than Upper boundary value is optional.</li><li>• 9 - Greater equal Upper boundary value is optional.</li></ul>	1	1
		This field is mandatory if LowerBoundaryLast Name is set.		
LowerBoundaryLast Name			40	0..1
UpperBoundaryLast Name			40	0..1
EmailAddressInter- val Cardinality: 0..un- bounded	IntervalBoundaryT yTypeCode		1	1
	LowerBoundaryEmail 1Address		241	0..1
	UpperBoundaryEmail 1Address		241	0..1

### Query Processing Conditions (QueryProcessingConditions)

The service nodes contain the service's business data.

Fields for the QueryProcessingConditions Node

Field	Description	Maximum Field Length	Cardinality
QueryHitsTotalNumberIndicator	The following values exist: <ul style="list-style-type: none"><li>• True</li><li>• False (default)</li></ul>		1
QueryHitsMaximumNumberValue	Enter the maximum number of hits. If no value is entered, the default is automatically set to 1000.	999999999	0..1
QueryHitsUnlimitedIndicator	The following values exist: <ul style="list-style-type: none"><li>• True</li><li>• False (default)</li></ul> Set <b>True</b> to get all data based on selection criteria.		1
QueryLastReturnedObjectID	The following values exist: <ul style="list-style-type: none"><li>• True</li><li>• False (default)</li></ul> If QueryHitsMaximumNumberValue is set and more data is available, you can set this value to <b>True</b> .		0..1

## Sample Payload

### ↔ Sample Code

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:aba="http://sap.com/xi/ABA">
    <soapenv:Header/>
    <soapenv:Body>
        <aba:BusinessUserSimpleByElementsQuery_sync>
            <BusinessUser>
                <PersonIDInterval>
                    <IntervalBoundaryTypeCode>1</IntervalBoundaryTypeCode>
                    <!--Optional:-->
                    <LowerBoundaryPersonID>9980035943</LowerBoundaryPersonID>
                    <!--Optional:-->
                </PersonIDInterval>
                <BusinessPartnerRoleCodeInterval>
                    <IntervalBoundaryTypeCode>1</IntervalBoundaryTypeCode>
                    <!--Optional:-->
                    <LowerBoundaryBusinessPartnerRoleCode>bup003</LowerBoundaryBusinessPartnerRoleCode>
                </BusinessPartnerRoleCodeInterval>
            </BusinessUser>
            <QueryProcessingConditions>
                <!--Optional:-->
                <QueryHitsMaximumNumberValue>1</QueryHitsMaximumNumberValue>
                <QueryHitsUnlimitedIndicator>false</QueryHitsUnlimitedIndicator>
            </QueryProcessingConditions>
        </aba:BusinessUserSimpleByElementsQuery_sync>
    </soapenv:Body>
</soapenv:Envelope>

```

```
</soapenv:Body>  
</soapenv:Envelope>
```

## Service Response

### Business User (BusinessUser)

#### ① Note

The fields below the node `User` will be filled.

Node or Field	Description	Maximum Field Length	Cardinality
PersonExternalID	Person External ID	60	0..1
PersonID	Person ID	10	1
PersonUUID	Person UUID	36	1
BusinessPartnerRoleCode	Business Partner Role Code	6	1
MarkedForArchivingIndicator	<ul style="list-style-type: none"><li>True</li><li>False</li></ul>		1
ValidityPeriod	Format: YYYY-MM-DD		1
StartDate			
Cardinality: 0..1			
EndDate	Format: YYYY-MM-DD		1
PersonalInformation	FormOfAddress	Form of address	4
Cardinality: 0..1	FirstName	First name	40
	LastName	Last name	40
	PersonFullName	Person full name	80
	AcademicTitle	Academic title	4
	CorrespondenceLanguage	Correspondence language	9
	MiddleName	Middle name	40

<b>Node or Field</b>	<b>Description</b>	<b>Length</b>	<b>Maximum Field Cardinality</b>
AdditionalLastName	Additional last name	40	0..1
BirthName	Birth name	40	0..1
NickName	Nick name	40	0..1
Initials	Initials	10	0..1
AcademicSecondTitle	Academic second title	4	0..1
LastNamePrefix	Last name prefix	4	0..1
LastNameSecondPrefix	Last name second prefix	4	0..1
NameSupplement	Name supplement	4	0..1
<b>User (only for Cloud)</b> Cardinality: 0..1	UserID	User ID	12
	UserName	User name/Alias	40
	LogonLanguageCode	Logon language	9

Node or Field	Description	Maximum Field Length	Cardinality
DateFormatCode	<p>The following values exist:</p> <ul style="list-style-type: none"> <li>• 1 - DD.MM.YYYY (Gregorian Date)</li> <li>• 2 - MM/DD/YYYY (Gregorian Date)</li> <li>• 3 - MM-DD-YYYY (Gregorian Date)</li> <li>• 4 - YYYY.MM.DD (Gregorian Date)</li> <li>• 5 - YYYY/MM/D D (Gregorian Date)</li> <li>• 6 - YYYY-MM-DD (Gregorian Date, ISO 8601)</li> <li>• 7 - GYY.MM.DD (Japanese Date)</li> <li>• 8 - GYY/MM/DD (Japanese Date)</li> <li>• 9 - GYY-MM-DD (Japanese Date)</li> <li>• A - YYYY/MM/D D (Islamic Date 1)</li> <li>• B - YYYY/MM/D D (Islamic Date 2)</li> </ul>	2	0..1

Node or Field	Description	Maximum Field Length	Cardinality
	<ul style="list-style-type: none"> <li>• C - YYYY/MM/D D (Iranian Date)</li> </ul>		
DecimalFormatCode	The following values exist: <ul style="list-style-type: none"> <li>• 1.234.567,89</li> <li>• X - 1,234,567,89</li> <li>• Y - 1 234 567,89</li> </ul>	2	0..1
TimeZoneCode	Time zone	10	0..1
TimeFormatCode	The following values exist: <ul style="list-style-type: none"> <li>• 0 - 24 Hour Format (Example: 12:05:10)</li> <li>• 1 - 12 Hour Format (Example: 12:05:10 PM)</li> <li>• 2 - 12 Hour Format (Example: 12:05:10 pm)</li> <li>• 3 - Hours from 0 to 11 (Example: 00:05:10 PM)</li> <li>• 4 - Hours from 0 to 11 (Example: 00:05:10 pm)</li> </ul>	2	0..1
LockedIndicator	Locked indicator	5	0..1

Node or Field	Description	Maximum Field Length	Cardinality
ValidityPeriod	Format: YYYY-MM-DD	1	
StartDate	Cardinality: 1  If no start date is maintained for the User, the StartDate for the BusinessUser is entered.		
EndDate	Format: YYYY-MM-DD  If no EndDate is maintained, it is set to 9999-12-31.	1	
Role	RoleName	Role name	40
Cardinality: 0..unbounded			1
UserAssignment	UserID	User ID	12
<b>(only for on-prem-use)</b>	UserName		40
	UserAssignmentStatusCode	The following values exist: <ul style="list-style-type: none"><li>• 1 - is reserved</li><li>• 2 - is assigned</li></ul>	0.1
WorkplaceInformation	EmailAddress	Email address	241
Cardinality: 0.1	PhoneInformation	PhoneType	Phone type
	Cardinality: 0.2	CountryDialingCode	Country dialing code
One set of phone information per phone type supported.	PhoneNumberAreaID	Phone number area code	10
	PhoneNumberSubscriberID	Phone number subscriber ID	30
PhoneNumberExtension	PhoneNumberExtension	Phone number extension	10
			0.1

Node or Field	Description	Maximum Field Length	Cardinality
FunctionalTitleName	Functional title name	40	0..1
Department	Department name	40	0..1
RoomNumber	Room number	10	0..1
Building	Building name	10	0..1

### Response Processing Conditions (ResponseProcessingConditions)

Field	Description	Maximum Field Length	Cardinality
HitsTotalNumberValue	Contains the number of users based on given criteria.	999999999	1
ReturnedQueryHitsNumberValue	Contains the number of found data sets for business users.	999999999	1
MoreHitsAvailableIndicator	The indicator is set if the query was limited to a number of hits, but more business user data sets are available based on the query.		1
LastReturnedObjectID	Displays the last row of the results list limited by the found hits or by the value given for QueryHitsMaximumNumber Value.	999999999	0..1

### Log (Log)

If errors occur, the log contains the information shown in the table below:

Field or Node	Description	Maximum Field Length	Cardinality
BusinessDocumentProcessingresultCode		2	0..1
MaximumLogItemSeverityCode	If several messages are stored for a business user, the most severe level is shown from the maximum of all received severity codes.	1	0..1

Field or Node		Description	Maximum Field Length	Cardinality
Item	TypeID	Message number	40	0..1
Cardinality: 0..unbounded	CategoryCode	Not in use	15	0..1
	SeverityCode	Severity code definition: <ul style="list-style-type: none"><li>• 1 - Information</li><li>• 2 - Warning</li><li>• 3 - Error</li></ul>	1	0..1
	Note	Contains the message texts.	200	1
	WebURI	Not in use		0..1

## Constraints

This service does not support:

- Freelancer (BBP010) business users

## Additional Information

### ⓘ Note

For more details about Communication Management, see [Communication Management \[page 2591\]](#).

## 4.2.9.5.3 Business User - Read Metadata

Technical name: `QueryBusinessUserMetadataIn`

This service enables you to read metadata information from your external data source such as an identity management system for service Business User with this synchronous inbound service.

This service provides the search parameter `RoleCategory`, which restricts the result set of the metadata information of the service. The response includes the metadata information based on the given criteria. If errors occur, the log contains information about the severity code, the message number and message texts.

This service is available on the SAP Business Accelerator Hub, for more information see [APIs on SAP Business Accelerator Hub](#).

## Service Request

### Service Nodes

The service nodes contain the service's business data.

Node or Field	Description	Cardinality
BusinessPartnerRoleCategoryInterval	IntervalBoundaryTypeCode You can only use the following value: <ul style="list-style-type: none"><li>• 1 - Equal</li></ul>	1..1
LowerBoundaryBusinessPartnerRoleCategoryCode	For example: BUP003	0..1

### Sample Payload

#### ↔ Sample Code

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
    xmlns:aba="http://sap.com/xi/ABA">  
    <soapenv:Header/>  
    <soapenv:Body>  
        <aba:BusinessUserMetaDataQuery_sync>  
            <!--Zero or more repetitions:-->  
            <BusinessPartnerRoleCategoryInterval>  
                <IntervalBoundaryTypeCode>1</IntervalBoundaryTypeCode>  
                <LowerBoundaryBusinessPartnerRoleCategoryCode>BUP003</LowerBoundaryBusinessPartnerRoleCategoryCode>  
            </BusinessPartnerRoleCategoryInterval>  
        </aba:BusinessUserMetaDataQuery_sync>  
    </soapenv:Body>  
</soapenv:Envelope>
```

## Service Response

Service Node	Description	Link to Details
BusinessUserMetaData	RoleCategoryDependentMetaData	This node contains all metadata, which depends on the role category of a business user, such as the role category with its role, external ID category with its external ID type or relationship category. <a href="#">RoleCategoryDependentMetaData [page 1083]</a>

Service Node	Description	Link to Details
CodeList	This node provides the available code lists for SAP specific codes. For example country/region code, academic title.	<a href="#">CodeList [page 1085]</a>
Log	This nodes displays occurred messages.	<a href="#">Log [page 1086]</a>

## Error Codes

Error Code	Message	Description
112	Interval Boundary Type Code &1 is not supported for BusinessPartnerRoleCategoryInterval.	You can only use the following value: • 1 - Equal
118	No data found by given search criteria.	To display all data, don't enter any value.

## Authentication Method

You can use the following authentication methods: User ID/password (Username Token), X.509 certificate (X509 Token) or Single Sign On using SAML (SAML Token).

## Constraints

Currently this SOAP service is only enabled for English.

## Additional Information

If you have any issues, report an incident for component CA-GTF-BUM.

### ⓘ Note

For more information about the API, choose the [Details](#) tab on the SAP Business Accelerator Hub.

## 4.2.9.5.3.1 RoleCategoryDependentMetaData

### Nodes and Fields

#### ⓘ Note

In the following table, field attributes are marked in blue.

RoleCategoryDependentMetaData

Node or Field	Description	Cardinality
BusinessPart nerRoleCategory	BusinessPartnerRoleCategoryCode For example: BUP003	1..1
Cardinality: 1..1	Description languageCode	For example: Employee 0..unbounded
BusinessPart nerRole	BusinessPartnerRoleCode For example: BBP005	1..1
Cardinality: 1..1	Description languageCode	0..unbounded
	DefaultIndicator	Can be <i>true</i> or <i>false</i>
BusinessPart nerExternalDCategory	BusinessPartnerExternalIDCategoryCode For example: HCM030	1..1
Cardinality: 0..1	Description languageCode	0..unbounded
BusinessPart nerExternalID	BusinessPartnerExternalIDCode For example: HCM030	1..1
Cardinality: 0..unbounded	Description languageCode	0..unbounded
	DefaultIndicator	The value can be either <i>true</i> or <i>false</i>
BusinessPart nerRelationshipCategory	BusinessPartnerRelationshipCategoryCode For example: BUR025	1..1
Cardinality: 0..1	Description languageCode	0..unbounded
Partner1_BusinessPartnerCategory	BusinessPartnerCategoryCode Can be:	1..1
Cardinality: 0..unbounded		<ul style="list-style-type: none"> <li>• 2 - Organization</li> <li>• 3 - Group</li> </ul>
	Description languageCode	0..unbounded
Partner2_BusinessPartnerCategory	BusinessPartnerCategoryCode Can be:	1..1
		<ul style="list-style-type: none"> <li>• 1 - Person</li> </ul>

Node or Field	Description	Description	Cardinality
Cardinality: 0..unbounded	languageCode		0..unbounded
NodeProperties	NodePath	For example: BUSINESS_USE R	1..1
Cardinality: 0..unbounded	NodeProperty	NodePropertyCode	Node property codes are:
	Cardinality: 0..unbounded	Description	<ul style="list-style-type: none"> <li>• 01 - enabled</li> <li>• 02 - disabled</li> <li>• 03 - read only</li> </ul>
FieldProperties	NodePath	For example:	1..1
0..unbounded		<ul style="list-style-type: none"> <li>• BUSINESS_USER-ROLE</li> <li>• BUSINESS_USER-VALIDITY_PERIOD</li> <li>• BUSINESS_USER_ASSIGNMENT</li> </ul>	
Fieldname		Name of a field.	1..1

Node or Field	Description	Cardinality
FieldProperty	FieldPropertyCode	Node property codes are:
Cardinality: 0..unbounded		1..1
		<ul style="list-style-type: none"> <li>• 01 - enabled</li> <li>• 02 - disabled</li> <li>• 03 - read only</li> <li>• 04 - mandatory</li> <li>• 05 - enabled, read only for update</li> <li>• 06 - mandatory, read only for update</li> </ul>
	Description	languageCode
		0..unbounded

#### ⓘ Note

To display more information about the service nodes on the SAP Business Accelerator Hub, select the [API Reference](#) tab and choose the [View Details](#) button in the table.

### 4.2.9.5.3.2 CodeList

#### Nodes and Fields

#### ⓘ Note

In the following table, field attributes are marked in blue.

CodeList

Node or Field	Description	Cardinality
FieldName	Name of the field.	1..1
FieldCodeList	This node provides the available code lists for SAP specific codes. For example coun-	1..1
FieldValue		
Cardinality: 0..unbounded		

Node or Field	Description	Cardinality
FieldDescription	languageCode	try/region code, academic title. 0..unbounded

**ⓘ Note**

To display more information about the service node on the SAP Business Accelerator Hub, select the operation.

### 4.2.9.5.3.3 Log

#### Nodes and Fields

**ⓘ Note**

In the following table, field attributes are marked in blue.

Log

Node or Field	Description	Cardinality
BusinessDocumentProcessingresultCode		0..1
MaximumLogItemSeverityCode	If several messages are stored for a business user, the most severe level is shown from the maximum of all received severity codes.	0..1
Item	TypeID	Message number.
Cardinality: 0..unbounded	CategoryCode	Not in use.
	SeverityCode	Severity code definition: <ul style="list-style-type: none"> <li>• 1 - Information</li> <li>• 2 - Warning</li> <li>• 3 - Error</li> </ul>
	Note	Contains the message texts.
	WebURI	Not in use.
		0..1

**ⓘ Note**

To display more information about the service node on the SAP Business Accelerator Hub, select the operation.

## 4.2.9.5.4 Integrating Business Role Change Documents

You can use this scenario to read the change documents of business roles.

### Context

This scenario allows you to read the change documents of business roles. We recommend that you limit the number of change documents according to the *Changed On* and the *Business Role ID* fields to avoid a high volume of data.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the *Maintain Communication Users* app.  
For more information, see the *Related information* section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the *Communication Systems* app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business Role Change Document Integration* scenario (SAP\_COM\_0366) in the *Communication Arrangements* app.

For more information, see the *Related information* section.

### Related Information

[How to Create a Communication Arrangement \[page 2596\]](#)

[How to Create Communication Users \[page 2594\]](#)

## 4.2.9.5.4.1 Business Role Changes - Read

Technical name: APS\_IAM\_API\_BROLE\_CDOC\_0001\_IWSG

This service enables you to read the change documents of business roles.

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the **Related Information** section.

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the service's business data.

Entity	Description	Necessity
BusinessRoleID	Business Role ID	Optional
ChangedOn	Change on Date/Time	Optional
ChangeCategory	Category of change  The following values are available: <ul style="list-style-type: none"><li>• <i>GE</i> = General changes to the business role attributes, such as the description</li><li>• <i>BU</i> = Assigned business users</li><li>• <i>BC</i> = Assigned business catalogs</li><li>• <i>FS</i> = Assigned launchpad spaces</li><li>• <i>RA</i> = Access restrictions</li><li>• <i>RV</i> = Restriction values</li><li>• <i>RS</i> = Restrictions</li></ul>	Optional
ChangeCategoryText	Category of change description (such as <i>Business catalog added</i> )	Optional
Attribute	Name of changed attribute	Optional
ValueChangedFrom	Old value	Optional
ValueChangedTo	New value	Optional
ChangedByUserName	Changed by user identified by user name	Optional
ChangedByUserID	Changed by user identified by user ID	Optional
ChangedByEmailAdress	Changed by user identified by email address	Optional
ChangedByGlobalUserID	Changed by user identified by <i>Global User ID</i>	Optional
Action	Action	Optional

## Service Response

The details of an API response are according to the operation types that are supported. For more information, see [Operation for Business Role Changes - Read \[page 1089\]](#).

Entity	Description
200	OK
400	Bad Request
403	Forbidden

## Additional Information

For more information about Communication Management, see the **Related Information** section.

## Related Information

[APIs on SAP Business Accelerator Hub](#)  
[Communication Management \[page 2591\]](#)

### 4.2.9.5.4.1.1 Operation for Business Role Changes - Read

The *Business Role Changes - Read* inbound service offers this operation:

Operation	HTTP Method	Sample URL
BusinessRoleChangeDocument	GET	GET<host>/sap/opu/odata/sap/APS_IAM_API_BROLE_CDOC/BusinessRoleChanges

## 4.2.9.5.5 Integrating Business User Change Documents

You can use this scenario to read the change documents of business users.

### Context

This scenario allows you to read the change documents of business users. We recommend that you limit the number of change documents according to the *Changed On* and the *Business User ID* fields to avoid a high volume of data.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the *Maintain Communication Users* app.  
For more information, see the *Related information* section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the *Communication Systems* app.  
For more information, see the *Related information* section.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business User Change Document Integration* scenario (SAP\_COM\_0327) in the *Communication Arrangements* app.  
For more information, see the *Related information* section.

### Related Information

[How to Create a Communication Arrangement \[page 2596\]](#)

[How to Create Communication Users \[page 2594\]](#)

## 4.2.9.5.5.1 Business User Changes - Read

Technical name: APS\_IAM\_API\_BUSER\_CDOC\_0001\_IWSG

This service enables you to read the change documents of business users.

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the *Related Information* section.

## Service Structure

### Service Header (optional)

The service header contains information about the service.

#### Entities

The entities contain the service's business data.

Entity	Description	Necessity
BusinessUserID	Business User ID	Optional
ChangedOn	Change on Date/Time	Optional
ChangeCategory	Category of change  The following values are available <ul style="list-style-type: none"><li>• <i>GE</i> = General changes to the business user attributes, such as the description</li><li>• <i>BR</i> = Business role assignments</li></ul>	Optional
ChangeCategoryText	Category of change description (such as <i>Business catalog added</i> )	Optional
Attribute	Name of changed attribute	Optional
ValueChangedFrom	Old value	Optional
ValueChangedTo	New value	Optional
ChangedByUserName	Changed by user identified by user name	Optional
ChangedByUserID	Changed by user identified by user ID	Optional
ChangedByUserEmailAddress	Changed by user identified by email address	Optional
ChangedByGlobalUserID	Changed by user identified by <i>Global User ID</i>	Optional
Action	Action	Optional

## Service Response

The details of an API response are according to the operation types that are supported. For more information, see [Operation for Business User Changes - Read \[page 1092\]](#).

Entity	Description
200	OK

Entity	Description
400	Bad Request
403	Forbidden

## Related Information

[APIs on SAP Business Accelerator Hub](#)  
[Communication Management \[page 2591\]](#)

### 4.2.9.5.5.1.1 Operation for Business User Changes - Read

The *Business User Changes - Read* inbound service offers the following operation:

Operation	HTTP Method	Sample URL
BusinessUserChangeDocument	GET	GET<host>/sap/opu/odata/sap/APS_IAM_API_BUSER_CDOC/BusinessUserChanges

### 4.2.9.5.6 Integrating Business Role Data (Read)

You can use this communication scenario to read business role data.

## Context

The communication scenario SAP\_COM\_OA04 allows you to read business role data, such as business role ID or description, read business roles, read assigned business users, launchpad spaces and business catalogs, or filter for specific combinations.

## Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the *Maintain Communication Users* app.

For more information, see the *Related information* section.

- Check if a communication system already exists for this scenario. If not, create a communication system in the *Communication Systems* app.
- Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business Role Read Integration* scenario (SAP\_COM\_OA04) in the *Communication Arrangements* app.

For more information, see the *Related information* section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.6.1 Business Role - Read Integration

This service enables you to do the following:

- Read business roles
- Read assigned business users, launchpad spaces and business catalogs
- Filter for specific combinations

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the *Related Information* section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_iam_api_brole_read	srvd_a2x	BusinessRole	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
BusinessRoles	Contains business roles.	Mandatory	Please refer to the <a href="#">Related Information</a> section.

## Additional Information

 Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

### [APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.6.1.1 Business Role Entity

Technical name: BusinessRoles

## Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
Description	Description	Optional
LongText	Long Text	Optional
BusinessRoleTemplateID	Business role template ID	Optional
IsLeading	Leading business role	Optional
LeadingBusinessRoleID	Leading business role ID	Optional
IsDerived	Derived business role	Optional
BusinessRoleGroup	Business role group	Optional
		Not available in SAP IBP

Property	Description	Necessity
ExposedToSAPBTP	Exposed to SAP BTP	Optional
PriceCategoryCode	Price category code	Optional
PriceCategory	Price category	Optional
WriteAccessRestrictionCode	<ul style="list-style-type: none"> <li>• <i>1: Unresticted</i></li> <li>• <i>2: Restricted</i></li> <li>• <i>3: No access</i></li> </ul>	Optional
ReadAccessRestrictionCode	<ul style="list-style-type: none"> <li>• <i>1: Unresticted</i></li> <li>• <i>2: Restricted</i></li> <li>• <i>3: No access</i></li> </ul>	Optional
F4AccessRestrictionCode	<ul style="list-style-type: none"> <li>• <i>1: Unresticted</i></li> <li>• <i>2: Restricted</i></li> <li>• <i>3: No access</i></li> </ul>	Optional
Imported	Imported	Optional
Exported	Exported	Optional
LastChangedDateTime	<i>Changed On:</i> <date>, <time>	Optional
LastChangedByUserID	<i>Changed By:</i> <user ID>	Optional
LastChangedByUserName	<i>Changed By:</i> <user name>	Optional
LastChangedByUserDescription	<i>Changed By:</i> <user description>	Optional
LastChangedByEMailAddress	<i>Changed By:</i> <email address>	Optional
LastChangedByGlobalUserID	<i>Changed By:</i> <global user ID>	

To display more information about the properties on the SAP Business Accelerator Hub, choose *Schema View*.

## Expandables

This entity is expandable with the following:

- Business Users
- Launchpad Spaces
- Business Catalogs
- Derived Business Roles
- Restrictions
  - RestrictionFields
  - RestrictionValues

## Related Information

[Business Role – Business User Entity \[page 1096\]](#)  
[Business Role – Launchpad Space Entity \[page 1096\]](#)  
[Business Role – Business Catalog Entity \[page 1097\]](#)  
[Business Role – Derived Business Role Entity \[page 1097\]](#)  
[Business Role - Restriction Entity \[page 1097\]](#)  
[Business Role – Restriction Field Entity \[page 1098\]](#)  
[Business Role – Restriction Value Entity \[page 1099\]](#)

### 4.2.9.5.6.1.1.1 Business Role – Business User Entity

Technical name: `BusinessRoleBusinessUsers`

#### Properties

Property	Description	Necessity
<code>BusinessRoleID</code>	Business role ID	Mandatory (Key)
<code>UserID</code>	Business user ID	Mandatory (Key)

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

### 4.2.9.5.6.1.1.2 Business Role – Launchpad Space Entity

Technical name: `BusinessRoleLaunchpadSpaces`

#### Properties

Property	Description	Necessity
<code>BusinessRoleID</code>	Business role ID	Mandatory (Key)
<code>SpaceID</code>	Space ID	Mandatory (Key)

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

### 4.2.9.5.6.1.1.3 Business Role – Business Catalog Entity

Technical name: BusinessRoleBusinessCatalogs

#### Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
BusinessCatalogID	Business catalog ID	Mandatory (Key)

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

### 4.2.9.5.6.1.1.4 Business Role – Derived Business Role Entity

Technical name: BusinessRoleDerivedRoles

#### Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
DerivedBusinessRoleID	Derived business role ID	Mandatory (Key)

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

### 4.2.9.5.6.1.1.5 Business Role - Restriction Entity

Technical name: BusinessRoleRestrictions

## Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
RestrictionTypeAssignmentUUID	Restriction type assignment UUID	Mandatory (Key)
AccessCategory	<i>W: Write</i> <i>R: Read</i> <i>F: Value Help</i>	Optional
RestrictionType	Restriction type	Optional

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.6.1.1.6 Business Role – Restriction Field Entity

Technical name: BusinessRoleRestrictionFields

## Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
RestrictionTypeAssignmentUUID	Restriction type assignment UUID	Mandatory (Key)
FieldName	Field name	Mandatory (Key)
MaintainStatus	<empty>: No maintenance status set. not_maint: Maintenance status set to <i>Not Maintained</i> . maint_derived: Maintenance Status set to <i>Maintained in Derived Role</i> .	Optional
IsLeading	Leading	Optional

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.6.1.1.7 Business Role – Restriction Value Entity

Technical name: BusinessRoleRestrictionValues

### Properties

Property	Description	Necessity
BusinessRoleID	Business role ID	Mandatory (Key)
RestrictionTypeAssignmentUUID	Restriction type assignment UUID	Mandatory (Key)
FieldName	Field name	Mandatory (Key)
RestrictionValueAssignmentUUID	Restriction value assignment UUID	Mandatory (Key)
LowValue	Low value	Optional
HighValue	High value	Optional

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.7 Integrating Business User Logon Details (Read)

You can use this communication scenario to read business user logon details.

### Context

The communication scenario SAP\_COM\_0889 allows you to read business user logon details, such as user name, validity, assigned business roles, read business users, read assigned business catalogs, business roles and application jobs, or filter for specific combinations.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.

- Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business User Logon Details Read Integration* scenario (SAP\_COM\_0889) in the *Communication Arrangements* app.

For more information, see the *Related information* section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.7.1 Business User Logon Details - Read Integration

This service enables you to

- Read business users
- Read assigned business catalogs, business roles and application jobs
- Filter for specific combinations

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the *Related Information* section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_iam_api_buser_read	srvd_a2x	BusinessUser	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
BusinessUsers	Contains Business Users	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more details about Communication Management, see [Communication Management \[page 2591\]](#).

## Related Information

[Business User Entity \[page 1101\]](#)

[APIs on SAP Business Accelerator Hub](#)

### 4.2.9.5.7.1.1 Business User Entity

Technical name: BusinessUsers

## Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
UserName	User name	Optional
Description	Description	Optional
FirstName	First name	Optional
LastName	Last name	Optional
EMailAddress	Email address	Optional
GlobalUserID	Global user ID	Optional

Property	Description	Necessity
BusinessUserGroup	Business user group	Optional Not available in SAP IBP
PriceCategoryCode	Price category code	Optional
PriceCategory	Price category	Optional
LockStatus	Lock status	Optional
Validity	Validity	Optional
IsLoggedIn	Is logged on	Optional
NeverLoggedIn	Never logged on	Optional
ValidFromDate	<i>Valid From</i> date	Optional
ValidToDate	<i>Valid To</i> date	Optional
LastLogonDateTime	Date and time of last logon	Optional
LastChangedDateTime	Date and time of last change	Optional
LastChangedByUserID	<i>Last Changed By</i> <user ID>	Optional
LastChangedByUserName	<i>Last Changed By</i> <user name>	Optional
LastChangedByUserDescription	<i>Last Changed By</i> <user description>	Optional
LastChangedByEmailAddress	<i>Last Changed By</i> <email address>	Optional
LastChangedByGlobalUserID	<i>Last Changed By</i> <global user ID>	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## Expandables

This entity is expandable with the following:

- Business Catalogs
- Business Roles
- Application Jobs

## Related Information

[Business User – Business Catalog Entity \[page 1103\]](#)

[Business User – Business Role Entity \[page 1103\]](#)

[Business User – Application Job Entity \[page 1103\]](#)

## 4.2.9.5.7.1.1.1 Business User – Business Catalog Entity

Technical name: BusinessUserBusinessCatalogs

### Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
BusinessCatalogID	Business catalog ID	Mandatory (Key)

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.7.1.1.2 Business User – Business Role Entity

Technical name: BusinessUserBusinessRoles

### Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
BusinessRoleID	Business role ID	Mandatory (Key)

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.7.1.1.3 Business User – Application Job Entity

Technical name: BusinessUserApplicationJobs

## Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
ApplicationJob	Application job	Mandatory (Key)

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.8 Integrating Support User Data (Read)

You can use this communication scenario to read support user data.

### Context

The communication scenario SAP\_COM\_0347 allows you to read support user request logs.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Support User Request Log Integration](#) scenario (SAP\_COM\_0347) in the [Communication Arrangements](#) app.

For more information, see the [Related information](#) section.

## 4.2.9.5.8.1 Support User - Read Integration

This service enables you to do the following:

- Read support users
- Read associated user information

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

### Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
api_iam_susr_read	srvd_a2x	SupportUser	0001

### Service Structure

#### Service Header (optional)

The service header contains information about the service.

#### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
SupportUsers	Contains support users.	Mandatory	Please refer to the <a href="#">Related Information</a> section.

### Related Information

#### APIs on SAP Business Accelerator Hub

[Communication Management \[page 2591\]](#)

## 4.2.9.5.8.1.1 Support User Entity

Technical name: `SupportUsers`

### Properties

Property	Description	Necessity
<code>CAMSupportUserRequestUUID</code>	Support user request UUID for SAP Cloud Access Manager.	Mandatory (Key)
<code>CAMSuppUserValdtyStartTime</code>	Support user validity start datetime for SAP Cloud Access Manager.	Optional
<code>CAMSupportUser</code>	Support user for SAP Cloud Access Manager.	Optional
<code>CAMSupportAccessLevel</code>	Support access level for SAP Cloud Access Manager.	Optional
<code>CAMSupportAccessCategory</code>	Support access category for SAP Cloud Access Manager.	Optional
<code>CAMSupportAccessCategoryText</code>	Support access category text for SAP Cloud Access Manager.	Optional
<code>CAMSuppAddedAuthorizationMode</code>	Support added authorization mode for SAP Cloud Access Manager.	Optional
<code>CAMSuppAddedAuthznModeText</code>	Support added authorization mode text for SAP Cloud Access Manager.	Optional
<code>CAMSuppAddedAuthorizationUser</code>	Support added authorization user for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncident</code>	Support incident for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncidentSystem</code>	Support incident system for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncidentSystemText</code>	Support incident system text for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncidentSystemNumber</code>	Support incident system number for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncidentTicketNumber</code>	Support incident ticket number for SAP Cloud Access Manager.	Optional
<code>CAMSupportIncidentTicketYear</code>	Support incident ticket year for SAP Cloud Access Manager.	Optional
<code>CAMSuppUserValdtyEndDateTime</code>	Support user validity end date and time for SAP Cloud Access Manager.	Optional

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## Expandables

This entity is expandable with the following:

- SupportUser

## Related Information

[Support User - User Entity \[page 1107\]](#)

### 4.2.9.5.8.1.1.1 Support User - User Entity

Technical name: SupportUserUsers

## Properties

Property	Description	Necessity
UserID	Contains the user ID.	Mandatory (Key)
UserDescription	Contains the user description.	Optional
IsTechnicalUser	Technical user.	Optional
AddressPersonID	Contains the person ID.	Optional
AddressID	Contains the address ID.	Optional

### 4.2.9.5.9 Integrating Communication Arrangement Data (Read)

You can use this communication scenario to read communication arrangement data.

## Context

This communication scenario allows you to read communication arrangements, read assigned inbound users, outbound users, inbound services and outbound services.

## Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Communication Arrangement Read Integration](#) scenario (SAP\_COM\_OA07) in the [Communication Arrangements](#) app.  
For more information, see the [Related information](#) section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.9.1 Communication Arrangement - Read Integration

This service enables you to:

- Read communication arrangements
- Read assigned inbound users, outbound users, inbound services and outbound services

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_api_ca_read	srvd_a2x	CommunicationArrangement	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
CommunicationArrangements	Contains communication arrangements	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Arrangements - Entity \[page 1109\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.9.1.1 Communication Arrangements - Entity

Technical name: CommunicationArrangements

### Properties

Property	Description	Necessity
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)

Property	Description	Necessity
CommunicationArrangementName	Communicaton arrangement name	Optional
CommunicationSystemID	Communication system ID	Optional
CommunicationSystemName	Communication system name	Optional
CommunicationScenarioID	Communication scenario ID	Optional
CommunicationScenarioName	Communication scenario name	Optional
LastChangedDateTime	Date and time of last change	Optional
LastChangedByUserID	Last changed by <user ID>	Optional
LastChangedByUserName	Last changed by <user name>	Optional
LastChangedByUserDescription	Last changed by <user description>	Optional
LastChangedByEMailAddress	Last changed by <email address>	Optional
LastChangedByGlobalUserID	Last changed by <global user ID>	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## Expandables

This entity is expandable with the following:

- [Inbound User](#)
- [Outbound User](#)
- [Inbound Services](#)
- [Outbound Services](#)

## Related Information

[Communication Arrangement – Inbound User Entity \[page 1111\]](#)

[Communication Arrangement – Outbound User Entity \[page 1111\]](#)

[Communication Arrangement – Inbound Service Entity \[page 1112\]](#)

[Communication Arrangement – Outbound Service Entity \[page 1113\]](#)

## 4.2.9.5.9.1.1.1 Communication Arrangement – Inbound User Entity

Technical name: CommArrangementsInboundUsers

### Properties

Property	Description	Necessity
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
UserID	User ID	Mandatory (Key)
UserName	User name	Optional
OAuth2ClientID	OAuth 2.0 client ID	Optional
OAuth2GrantType	OAuth 2.0 grant type	Optional
AuthMethod	<b>x509</b> : SSL Client Certificate <b>basic</b> : User ID and Password <b>none</b> : None <b>oauth2</b> : OAuth 2.0 <b>oauth1</b> : OAuth 1.0 princ_prop: Principal Propagation oauth2_mtls: OAuth 2.0 (mTLS)	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.9.1.1.2 Communication Arrangement – Outbound User Entity

Technical name: CommArrangementsOutboundUsers

## Properties

Property	Description	Necessity
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
UserName	User name	Mandatory (Key)
OAuth2ClientID	OAuth 2.0 client ID	Mandatory (Key)
CertificateID	Certificate ID	Optional
AuthMethod	<b>x509</b> : SSL Client Certificate <b>basic</b> : User ID and Password <b>none</b> : None <b>oauth2</b> : OAuth 2.0 - <b>oauth1</b> : OAuth 1.0 <b>princ_prop</b> : Principal Propagation <b>oauth2_mtls</b> : OAuth 2.0 (mTLS)	Optional

### ① Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.9.1.1.3 Communication Arrangement – Inbound Service Entity

Technical name: CommArrangementsInboundServ

## Properties

Property	Description	Necessity
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
ServiceUUID	Service UUID	Mandatory (Key)
ServiceNumber	Service number	Optional

Property	Description	Necessity
ServiceID	Service ID	Optional
ServiceType	Service type	Optional
IsHidden	Hidden	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.9.1.1.4 Communication Arrangement – Outbound Service Entity

Technical name: CommArrangementsOutboundServ

### Properties

Property	Description	Necessity
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
ServiceUUID	Service UUID	Mandatory (Key)
ServiceNumber	Service number	Optional
ServiceID	Service ID	Optional
ServiceType	Service type	Optional
Status	Status	Optional
IsActive	Active	Optional
URL	URL	Optional
TCPPort	TCP port	Optional
SOAPWSRMVersion	SOAPWSRM version	Optional
JobTimezoneCode	Job time zone code	Optional

Property	Description	Necessity
JobStartTime	Job start time	Optional
JobPackageSize	Job package size	Optional
JobRecurrenceValue	Job recurrence value	Optional
JobFrequency	Job frequency	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.10 Integrating Communication Management Data (Read and Write)

You can use this communication scenario to read and write communication arrangement, communication system and communication user data.

### Context

This communication scenario allows you to read and write communication arrangements, communication systems and communication users.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Communication Management Integration](#) scenario (SAP\_COM\_OA48) in the [Communication Arrangements](#) app.

For more information, see the [Related information](#) section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.10.1 Communication Arrangement - Read and Write Integration

This service enables you to:

- Read and write communication arrangements
- Read and write assigned inbound users, outbound users, inbound services and outbound services

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_ca_a4c_odata	srvd_a2x	aps_com_ca_a4c_odata	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity
CommunicationArrangements	Contains communication arrangement root data	Mandatory
Properties	Contains communication arrangement root property names	Optional

Entity	Description	Necessity
PropertyValues	Contains communication arrangement root property values	Optional
InboundUsers	Contains the reference to the communication system inbound user	Optional
InboundServices	Contains the communication arrangement inbound service data	Optional
InboundServiceProperties	Contains communication arrangement inbound service property names	Optional
InboundServicePropertyValue	Contains communication arrangement inbound service property values	Optional
OutboundUsers	Contains the reference to the communication system outbound user	Optional
OutboundUserAuthScopes	Contains additional scopes for outbound users	Optional
OutboundServices	Contains the communication arrangement outbound service data	Optional
OutboundServiceProperties	Contains communication arrangement outbound service property names	Optional
OutboundServicePropertyValue	Contains communication arrangement outbound service property values	Optional

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Arrangements - Entity \[page 1109\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.10.2 Communication System - Read and Write Integration

This service enables you to:

- Read and write communication systems
- Read and write assigned inbound users, outbound users and communication arrangements

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

### Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_cs_a4c_odata	srvd_a2x	aps_com_cs_a4c_odata	0001

### Service Structure

#### Service Header (optional)

The service header contains information about the service.

#### Entities

The entities contain the business data of the service.

Entity	Description	Necessity
InboundUsers	Contains references to the communication user ID	Mandatory
OutboundUsers	Contains outbound user data	Optional
BusinessPartners	Contains references to the business partner IDs	Optional

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Arrangements - Entity \[page 1109\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.10.3 Communication User - Read and Write Integration

This service enables you to:

- Read and write communication users
- Read and write assigned certificates, communication systems and communication arrangements

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_cu_a4c_odata	srvd_a2x	aps_com_cu_a4c_odata	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity
CommunicationUsers	Contains communication user root data	Mandatory
Certificates	Contains the assigned certificates	Optional

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Arrangements - Entity \[page 1109\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.11 Integrating Communication System Data (Read)

You can use this communication scenario to read communication system data.

## Context

This communication scenario allows you to read communication system data, such as read communication systems, read assigned inbound users, outbound users and communication arrangements.

## Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.

- Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Communication System Read Integration* scenario (SAP\_COM\_0A06) in the *Communication Arrangements* app.

For more information, see the *Related information* section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.11.1 Communication System - Read Integration

This service enables you to:

- Read communication systems
- Read assigned inbound users, outbound users and communication arrangements

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the *Related Information* section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_api_cs_read	srvd_a2	CommunicationSystem	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
CommunicationSystems	Contains communication systems	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Systems - Entity \[page 1121\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.11.1.1 Communication Systems - Entity

Technical name: CommunicationSystems

## Properties

Property	Description	Necessity
CommunicationSystemID	Communication system ID	Mandatory (Key)
CommunicationSystemName	Communication system name	Optional
HostName	Host name	Optional
UIHostName	UI host name	Optional
TCPPort	TCP port	Optional
LogicalSystemID	Logical system ID	Optional

Property	Description	Necessity
BusinessSystemID	Business system ID	Optional
InboundOnly	Inbound only	Optional
HubSystem	Hub system	Optional
SAPClient	SAP client	Optional
RFCLoadBalancing	RFC load balancing	Optional
RFCSAPSystemID	RFC SAP system ID	Optional
RFCSAPSystemNumber	RFC SAP system ID number	Optional
RFCLogonGroup	RFC logon group	Optional
RFCMessageServerTargetHost	RFC message server target host	Optional
OwnSystem	Own system	Optional
OAuth2IdentityProviderActive	OAuth 2.0 identity provider is active	Optional
OAuth2IdentityProviderName	OAuth 2.0 identity provider name	Optional
OAuth2IDPUserLogonType	OAuth 2.0 IDP user logon type	Optional
OAuth2AuthEndpoint	OAuth 2.0 authorization endpoint	Optional
OAuth2TokenEndpoint	OAuth 2.0 token endpoint	Optional
OAuth2Audience	OAuth 2.0 audience	Optional
SAPCloudConnectorActive	SAP Cloud Connector is active	Optional
SAPCloudConnectorLocationID	SAP Cloud Connector location ID	Optional
SAMLBearerAssertionProvActive	SAML Bearer Assertion Provider is active	Optional
SAMLBearerAssertionProvName	SAML Bearer Assertion Provider name	Optional
SAMLBearerAssProvUserLogonType	SAML Bearer Assertion Provider user logon type	Optional
OwnerContactPersonName	Owner contact person name	Optional
OwnerContactPersonPhone	Owner contact person phone	Optional
OwnerContactPersonEMail	Owner contact person email	Optional
LastChangedDateTime	Date and time of last change	Optional

Property	Description	Necessity
LastChangedByUserID	Last changed by <user ID>	Optional
LastChangedByUserName	Last changed by <user name>	Optional
LastChangedByUserDescription	Last changed by <user description>	Optional
LastChangedByEMailAddress	Last changed by <email address>	Optional
LastChangedByGlobalUserID	Last changed by <global user ID>	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## Expandables

This entity is expandable with the following:

- Inbound Users
- Outbound Users
- Communication Arrangements

## Related Information

[Communication System – Inbound User Entity \[page 1123\]](#)

[Communication System – Outbound User Entity \[page 1124\]](#)

[Communication System – Communication Arrangement Entity \[page 1125\]](#)

## 4.2.9.5.11.1.1.1 Communication System – Inbound User Entity

Technical name: `CommSystemInboundUsers`

## Properties

Property	Description	Necessity
CommunicationSystemID	Communication system ID	Mandatory (Key)
InboundUUID	Inbound UUID	Mandatory (Key)
UserID	User ID	Mandatory (Key)
UserName	User name	Optional
OAuth2ClientID	OAuth 2.0 client ID	Optional
OAuth2GrantType	OAuth 2.0 grant type	Optional
AuthMethod	<b>x509</b> : SSL Client Certificate <b>basic</b> : User ID and Password <b>none</b> : None <b>oauth2</b> : OAuth 2.0 <b>oauth1</b> : OAuth 1.0 <b>princ_prop</b> : Principal Propagation <b>oauth2_mtls</b> : OAuth 2.0 (mTLS)	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.11.1.1.2 Communication System – Outbound User Entity

Technical name: CommSystemOutboundUsers

## Properties

Property	Description	Necessity
CommunicationSystemID	Communication system ID	Mandatory (Key)

Property	Description	Necessity
OutboundUUID	Outbound UUID	Mandatory (Key)
UserName	User name	Mandatory (Key)
OAuth2ClientID	OAuth 2.0 client ID	Mandatory (Key)
CertificateID	Certificate ID	Optional
AuthMethod	<p><b>x509</b>: SSL Client Certificate</p> <p><b>basic</b>: User ID and Password</p> <p><b>none</b>: None</p> <p><b>oauth2</b>: OAuth 2.0</p> <p><b>oauth1</b>: OAuth 1.0</p> <p><b>princ_prop</b>: Principal Propagation</p> <p><b>oauth2_mtls</b>: OAuth 2.0 (mTLS)</p>	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

### 4.2.9.5.11.1.1.3 Communication System – Communication Arrangement Entity

Technical name: CommSystemCommArrangements

#### Properties

Property	Description	Necessity
CommunicationSystemID	Communication system ID	Mandatory (Key)
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
CommunicationArrangementName	Communication arrangement name	Optional
CommunicationScenarioid	Communication scenario ID	Optional

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.12 Integrating Communication User Data (Read)

You can use this communication scenario to read communication user data.

### Context

This communication scenario allows you to read communication user data, such as read communication users, read assigned certificates, communication systems and communication arrangements.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.

For more information, see the [Related information](#) section.

2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Communication User Read Integration](#) scenario (SAP\_COM\_0A05) in the [Communication Arrangements](#) app.

For more information, see the [Related information](#) section.

### Related Information

[Communication Management \[page 2591\]](#)

## 4.2.9.5.12.1 Communication User - Read Integration

This service enables you to:

- Read communication users

- Read assigned certificates, communication systems and communication arrangements

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_com_api_cusr_read	srvd_a2	CommunicationUser	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
CommunicationUsers	Contains communication users	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Communication Users - Entity \[page 1128\]](#)

[APIs on SAP Business Accelerator Hub](#)

## 4.2.9.5.12.2 Communication Users - Entity

Technical name: CommunicationUsers

### Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
UserName	User Name	Optional
Description	Description	Optional
LastLogonDateTime	Date and time of last logon	Optional
LockStatus	Lock status	Optional
PasswordStatus	Password status	Optional
LastChangedDateTime	Date and time of last change	Optional
LastChangedByUserID	Last changed by <user ID>	Optional
LastChangedByName	Last changed by <user name>	Optional
LastChangedByUserDescription	Last changed by <user description>	Optional
LastChangedByEmail	Last changed by <email address>	Optional
LastChangedByGlobalUserID	Last changed by <global user ID>	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## Expandables

This entity is expandable with the following:

- Certificates
- Communication Systems
- Communication Arrangements

## Related Information

[Communication User - Certificate Entity \[page 1129\]](#)

[Communication User – Communication Arrangement Entity \[page 1130\]](#)

[Communication User – Communication System Entity \[page 1130\]](#)

## 4.2.9.5.12.2.1 Communication User - Certificate Entity

Technical name: `CommunicationUserCertificates`

## Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
SubjectHash	Subject Hash	Mandatory (Key)
IssuerHash	Issuer Hash	Mandatory (Key)
Issuer	Issuer	Optional
Subject	Subject	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.12.2.2 Communication User – Communication Arrangement Entity

Technical name: CommUserCommArrangements

### Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
CommunicationArrangementUUID	Communication arrangement UUID	Mandatory (Key)
CommunicationArrangementName	Communication arrangement name	Optional
CommunicationSystemID	Communication system ID	Optional
CommunicationScenarioID	Communication scenario ID	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.12.2.3 Communication User – Communication System Entity

Technical name: CommUserCommSystems

### Properties

Property	Description	Necessity
UserID	User ID	Mandatory (Key)
CommunicationSystemID	Communication system ID	Mandatory (Key)

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose [Schema View](#).

## 4.2.9.5.13 Integrating Content Security Policy Data (Read)

You can use this communication scenario to read Content Security Policy (CSP) data.

### Context

This communication scenario allows you to read Content Security Policy (CSP) data, such as read Content Security Policy statuses, or read trusted sites.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Content Security Policy Read Integration](#) scenario (SAP\_COM\_0A08) in the [Communication Arrangements](#) app.  
For more information, see the [Related information](#) section.

### Related Information

[Communication Management \[page 2591\]](#)

## 4.2.9.5.13.1 Content Security Policy - Read Integration

This service enables you to

- Read Content Security Policy statuses

- Read trusted sites

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_sec_api_csp_read	srvd_a2x	ContentSecurityPolicy	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
ContentSecurityPolicyStatus	Contains the Content Security Policy statuses	Mandatory	For more information, see the <a href="#">Related Information</a> section.
TrustedSites	Contains trusted sites	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Content Security Policy Status - Entity \[page 1133\]](#)

[Trusted Sites - Entity \[page 1133\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

### 4.2.9.5.13.1.1 Content Security Policy Status - Entity

Technical name: ContentSecurityPolicyStatus

#### Properties

Property	Description	Necessity
ContentSecurityPolicyStatus	Content Security Policy Status	Mandatory (Key)

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

### 4.2.9.5.13.1.2 Trusted Sites - Entity

Technical name: TrustedSites

#### Properties

Property	Description	Necessity
ListName	Trusted Sites	Mandatory (Key)

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## Expandables

This entity is expandable with:

- TrustedContent

## Related Information

[Trusted Sites – Content Entity \[page 1134\]](#)

## 4.2.9.5.13.1.2.1 Trusted Sites – Content Entity

Technical name: TrustedSiteContent

## Properties

Property	Description	Necessity
ListName	Trusted Sites	Mandatory (Key)
TrustedSite	Trusted Site	Mandatory (Key)

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.14 Integrating Protection Allowlist Data (Read)

You can use this communication scenario to read protection allowlist data.

## Context

This communication scenario allows you to read protection allowlist data, such as read clickjacking protection data, read trusted network zones, read trusted CSS Style Sheets, read Cross-Origin Resource Sharing.

## Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Protection Allowlist Read Integration](#) scenario (SAP\_COM\_OA09) in the [Communication Arrangements](#) app.  
For more information, see the [Related information](#) section.

## Related Information

[Communication Management \[page 2591\]](#)

### 4.2.9.5.14.1 Protection Allowlist - Read Integration

This service enables you to:

- Read clickjacking protection data
- Read trusted network zones
- Read trusted CSS Style Sheets
- Read Cross-Origin Resource Sharing

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space if It Exists)	Repository ID	Service Name (incl. Name-space if It Exists)	Version
aps_sec_api_pal_read	srvd_a2	ProtectionAllowlist	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
ClickjackingProtection	Contains trusted hosts for Clickjacking Protection	Mandatory	For more information, see the <a href="#">Related Information</a> section.
TrustedNetworkZones	Contains trusted hosts for trusted network zones	Mandatory	For more information, see the <a href="#">Related Information</a> section.
TrustedCSSStyleSheets	Contains trusted hosts CSS Style Sheets	Mandatory	For more information, see the <a href="#">Related Information</a> section.
CrossOriginResourceSharing	Contains trusted hosts for Cross-Origin Resource sharing	Mandatory	For more information, see the <a href="#">Related Information</a> section.

## Additional Information

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[Clickjacking Protection - Entity \[page 1137\]](#)

[CORS - Entity \[page 1137\]](#)

[APIs on SAP Business Accelerator Hub](#)

[Communication Management \[page 2591\]](#)

## 4.2.9.5.14.1.1 Clickjacking Protection - Entity

Technical name: ClickjackingProtection

### Properties

Property	Description	Necessity
HostName	Host name	Mandatory: Key
SchemeCode	Scheme code	Optional
Port	Port	Optional

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.14.1.2 CORS - Entity

Technical name: CrossOriginResourceSharing

### Properties

Property	Description	Necessity
HostName	Host Name	Mandatory: Key
ServicePath	Service Path	Optional
HTTPGet	HTTP Get	Optional
HTTPPost	HTTP Post	Optional
HTTPHead	HTTP head	Optional
HTTPPut	HTTP Put	Optional
HTTPPatch	HTTP Patch	Optional
HTTPDelete	HTTP Delete	Optional
HTTPTrace	HTTP Trace	Optional

Property	Description	Necessity
HTTPOptions	HTTP Options	Optional
HTTPConnect	HTTP Connect	Optional
AllowCredentials	Allow Credentials	Optional
AllowPrivateNetworkAccess	Allow Private Network Access	Optional
MaximumAge	MaximumAge	Optional

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## Expandables

This entity is expandable with the following:

- Allowed Headers
- Exposed Headers

## Related Information

[CORS – Allowed Header Entity \[page 1138\]](#)

[CORS – Exposed Header Entity \[page 1139\]](#)

## 4.2.9.5.14.1.2.1 CORS – Allowed Header Entity

Technical name: CORSAllowedHeaders

## Properties

Property	Description	Necessity
HostName	Host Name	Mandatory: Key
AllowedHeader	Allowed header	Mandatory (Key)

**ⓘ Note**

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.14.1.2.2 CORS – Exposed Header Entity

Technical name: CORSExposedHeaders

## Properties

Property	Description	Necessity
HostName	Host Name	Mandatory: Key
ExposedHeader	Exposed header	Mandatory (Key)

**ⓘ Note**

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.14.1.3 Trusted CSS Style Sheets - Entity

Technical name: TrustedCSSstyleSheets

## Properties

Property	Description	Necessity
HostName	Host name	Mandatory (Key)
SchemeCode	Scheme code	Optional
Port	Port	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.14.1.4 Trusted Network Zones - Entity

Technical name: TrustedNetworkZones

## Properties

Property	Description	Necessity
HostName	Host name	Mandatory (Key)
SchemeCode	Scheme code	Optional
Port	Port	Optional

### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15 Integrating Certificate Data (Read)

### Context

You can use this communication scenario to read certificate data.

The communication scenario SAP\_COM\_0A13 allows you to read certificate data, such as client certificates and certificate trust lists.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related Information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [Certificate Read Integration](#) scenario (SAP\_COM\_0A13) in the [Communication Arrangements](#) app.

For more information, see the [Related Information](#) section.

### Related Information

[Communication Management \[page 2591\]](#)

## 4.2.9.5.15.1 Certificates - Read Integration

This service enables you to:

- Read client certificates
- Read certificate trust lists
- Read signing certificates

This service is published on the SAP Business Accelerator Hub. For more information about APIs, see the [Related Information](#) section.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

Service Group (incl. Name-space If It Exists)	Repository ID	Service Name (incl. Name-space If It Exists)	Version
aps_sec_api_cert_read	srvd_a2x	Certificates	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
ClientCertificates	Contains client certificates	Mandatory	For more information, see the <a href="#">Related Information</a> section.
CertificateTrustList	Contains certificate trust lists	Mandatory	For more information, see the <a href="#">Related Information</a> section.
SigningCertificates	Contains signing certificates	Mandatory	For more information, see the <a href="#">Related Information</a> section.

### ⓘ Note

For more information about Communication Management, see the [Related Information](#) section.

## Related Information

[APIs on SAP Business Accelerator Hub](#)  
[Communication Management \[page 2591\]](#)

## 4.2.9.5.15.1.1 Client Certificates - Entity

Technical name: ClientCertificates

## Properties

Property	Description	Necessity
Certificate	Certificate	Mandatory (Key)
Description	Description	Optional
ManagedBy	Managed by	Optional
IssuedTo	Issued to	Optional
IssuedBy	Issue by	Optional
PublicKey	Public key	Optional
PublicKeyAlgorithm	Public key algorithm	Optional
FingerPrint	Fingerprint	Optional
SerialNumber	Serial number	Optional
SignatureAlgorithm	Signature algorithm	Optional
Status	Certificate status	Optional
StatusText	Certificate status text	Optional
IsValid	Valid	Optional
ValidFromDateTime	<b>Valid From</b> date and time	Optional
ValidToDateDateTime	<b>Valid To</b> date and time	Optional
LastChangedDateTime	Date and time of last change	Optional
LastChangedByUserID	<i>Last Changed By</i> <user ID>	Optional
LastChangedByUserName	<i>Last Changed By</i> <user name>	Optional
LastChangedByUserDescription	<i>Last Changed By</i> <user description>	Optional
LastChangedByEMailAddress	<i>Last Changed By</i> <email address>	Optional
LastChangedByGlobalUserID	<i>Last Changed By</i> <global user ID>	Optional

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## Expandables

This entity is expandable with the following:

- Subject
- Issuer
- CommunicationSystems

## Related Information

[Client Certificates - Subject \[page 1144\]](#)

[Client Certificates - Issuer \[page 1145\]](#)

[Client Certificates - Communication Systems \[page 1145\]](#)

## 4.2.9.5.15.1.1.1 Client Certificates - Subject

Technical name: ClientCertificatesSubject

## Properties

Property	Description	Necessity
Certificate	Certificate	Mandatory (Key)
IssuedTo	Issued to	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15.1.1.2 Client Certificates - Issuer

Technical name: ClientCertificatesIssuer

### Properties

Property	Description	Necessity
Certificate	Certificate	Mandatory (Key)
IssuedBy	Issued by	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15.1.1.3 Client Certificates - Communication Systems

Technical name: ClientCertificatesCommSystems

### Properties

Property	Description	Necessity
Certificate	Certificate	Mandatory (Key)
CommunicationSystemID	Communication system ID	Mandatory (Key)

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15.1.2 Certificate Trust List - Entity

Technical name: CertificateTrustList

### Properties

Property	Description	Necessity
TrustListKey	Trust list key	Mandatory (Key)
ManagedBy	Managed by	Optional
IssuedTo	Issued to	Optional
IssuedBy	Issued by	Optional
PublicKey	Public key	Optional
PublicKeyAlgorithm	Public key algorithm	Optional
FingerPrint	Fingerprint	Optional
SerialNumber	Serial number	Optional
SignatureAlgorithm	Signature algorithm	Optional
IsValid	Valid	Optional
ValidFromDateTime	<b>Valid From</b> date and time	Optional
ValidToDateDateTime	<b>Valid To</b> date and time	Optional

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

### Expandable

This entity is expandable with the following:

- Subject
- Issuer

## Related Information

[Certificate Trust List - Subject \[page 1147\]](#)

[Certificate Trust List - Issuer \[page 1147\]](#)

### 4.2.9.5.15.1.2.1 Certificate Trust List - Subject

Technical name: CertificateTrustListSubject

#### Properties

Property	Description	Necessity
TrustListKey	Trust list key	Mandatory (Key)
IssuedTo	Issued to	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

#### ⓘ Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

### 4.2.9.5.15.1.2.2 Certificate Trust List - Issuer

Technical name: CertificateTrustListIssuer

## Properties

Property	Description	Necessity
TrustListKey	Trust list key	Mandatory (Key)
IssuedBy	Issued by	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15.1.3 Signing Certificates - Entity

Technical name: `SigningCertificates`

## Properties

Property	Description	Necessity
SigningKey	Signing key	Mandatory (Key)
CertificateType	Certificate type	Optional
IssuedTo	Issued to	Optional
IssuedBy	Issued by	Optional
PublicKey	Public key	Optional
PublicKeyAlgorithm	Public key algorithm	Optional
FingerPrint	Fingerprint	Optional
SerialNumber	Serial number	Optional

Property	Description	Necessity
SignatureAlgorithm	Signature algorithm	Optional
IsValid	Valid	Optional
ValidFromDateTime	<b>Valid From</b> date and time	Optional
ValidToDateTime	<b>Valid To</b> date and time	Optional
CertificateTypeName	Certificate type name	Optional

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## Expandables

This entity is expandable with the following:

- Subject
- Issuer
- CommunicationSystems

## Related Information

[Signing Certificates - Subject \[page 1149\]](#)

[Signing Certificates - Issuer \[page 1150\]](#)

[Signing Certificates - Communication Systems \[page 1151\]](#)

## 4.2.9.5.15.1.3.1 Signing Certificates - Subject

Technical name: `SigningCertificatesSubject`

## Properties

Property	Description	Necessity
SigningKey	Signing key	Mandatory (Key)

Property	Description	Necessity
IssuedTo	Issued to	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

## 4.2.9.5.15.1.3.2 Signing Certificates - Issuer

Technical name: SigningCertificatesIssuer

### Properties

Property	Description	Necessity
IssuedBy	Issued by	Mandatory (Key)
OrganizationUnit	Organizational unit	Optional
Organization	Organization	Optional
Country	Country	Optional
City	City	Optional
State	State	Optional

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

### 4.2.9.5.15.1.3.3 Signing Certificates - Communication Systems

Technical name: `SigningCertificatesCommSystems`

#### Properties

Property	Description	Necessity
<code>SigningKey</code>	Signing key	Mandatory (Key)
<code>CommunicationSystemID</code>	Communication system ID	Mandatory (Key)

#### Note

To display more information about the properties on the SAP Business Accelerator Hub, choose **Schema View**.

### 4.2.9.5.16 Integrating Cross-domain Identity Management

You can use this communication scenario to maintain data via SCIM.

#### Context

The communication scenario SAP\_COM\_0465 allows you to maintain data via SCIM (System for Cross-domain Identity Management).

#### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the [Maintain Communication Users](#) app.  
For more information, see the [Related information](#) section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the [Communication Systems](#) app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the [System for Cross-domain Identity Management Integration](#) scenario (SAP\_COM\_0465) in the [Communication Arrangements](#) app.

For more information, see the [Related information](#) section.

## Related Information

[Communication Management \[page 2591\]](#)

[SCIM Interface for IAM \[page 1152\]](#)

### 4.2.9.5.16.1 SCIM Interface for IAM

SCIM stands for System for Cross-domain Identity Management. It is an open standard that allows for the automation of user provisioning and management in cloud-based applications and services.

You can use this interface provided by the communication scenario SAP\_COM\_0465 to maintain and retrieve business users in your Cloud system and assign business roles and business user groups to them.

For more information about SCIM in general, see the [Related Information](#) section.

#### ⓘ Note

The write operations are protected with a X-CSRF-Token.

## SCIM Extensions

In addition, your Cloud system provides the following extensions to extend the endpoints of users and groups:

**Extension name:** [urn:ietf:params:scim:schemas:extension:sap:2.0:User](#)

The following singular attributes are defined:

- *validFrom*: Valid From date of a business user
- *validTo*: Valid To date of a business user
- *loginTime*: Last login time of a business user
- *userUuid*: Defines the global user ID
- *userUuidHistory*: All global user IDs that were assigned to the business user
  - *value*: Global user ID assigned to a business user
  - *active*: Global user ID currently actively in use

Example

#### ↴ Sample Code

```
"urn:ietf:params:scim:schemas:extension:sap:2.0:User" : {
    "validFrom" : "2023-05-01T00:00:00Z",
    "validTo" : "2123-05-01T00:00:00Z",
    "loginTime" : "2024-01-01T11:22:33Z",
    "userUuid" : "<GLOBAL_USER_ID>",
    "userUuidHistory" : [
        {
            "value" : "<GLOBAL_USER_ID>",
            "active" : true
        }
    ]
}
```

```

        },
        {
            "value" : "<GLOBAL_USER_ID_OLD>",
            "active" : false
        }
    ]
}

```

**Extension name:** [urn:ietf:params:scim:schemas:extension:sap.odm:2.0:User](#)

The following singular attributes are defined:

- [workAssignment](#): Contains employee information
  - [id](#): Business partner ID
  - [startDate](#): Valid From date of an employee
  - [endDate](#): Valid To date of an employee
  - [primary](#): Employee is primary

#### ↔ Sample Code

```
"urn:ietf:params:scim:schemas:extension:sap.odm:2.0:User" : {
    "workAssignment" : [
        {
            "id" : "0000000001",
            "startDate" : "2000-01-01T00:00:00Z",
            "endDate" : "9999-12-31T23:59:59Z",
            "primary" : true
        }
    ]
}
```

**Extension name:** [urn:ietf:params:scim:schemas:extension:sap:2.0:Group](#)

The following singular attributes are defined:

- [type](#): Describes whether the group is a business role ([authorization](#)) or a business user group ([userGroup](#))
- [supportedOperations](#): Describes the possible operations that can be performed either with the group [membership](#) or [readWrite](#)

Example

#### ↔ Sample Code

```
"urn:ietf:params:scim:schemas:extension:sap:2.0:Group": {
    "type": "authorization"
    "supportedOperations": "membership"
}
```

## Parameter Equivalents in your Cloud system

### User

Parameter in SCIM Standard	Equivalent in your Cloud System
<i>id</i>	Unique <i>business user ID</i>
<i>externalId</i>	<i>worker/employee ID</i>
<i>userName</i>	<i>user name</i>
<i>groups["type"]</i>	Determines whether it is a business role or business user group.

## Group

Parameter in SCIM Standard	Equivalent in your Cloud System
<i>id</i>	Unique internal ID for SCIM containing a prefix ( <i>BROL_</i> or <i>BUGR_</i> ) and the business role or business user group ID to ensure that no duplicates exist. This ID has no corresponding Cloud equivalent.
<i>externalId</i>	Business role or business user group ID

## Related Information

[System for cross-domain Identity Management](#) ↗

[Example: User Endpoint \[page 1154\]](#)

[Example: Group Endpoint \[page 1155\]](#)

## 4.2.9.5.16.2 Example: User Endpoint

Please find an example of a user endpoint below:

### ↔ Sample Code

```
{
  "schemas" : [
    "urn:ietf:params:scim:schemas:core:2.0:User",
    "urn:ietf:params:scim:schemas:extension:sap:2.0:User"
  ],
  "id" : "CB0000000001",
  "externalId" : "WORKER_ID",
  "userName" : "JOHN_DOE",
  "name" : {
    "formatted" : "John Doe",
    "familyName" : "Doe",
    "givenName" : "John"
  },
  "displayName" : "John Doe",
  "emails" : [
    {
      "value" : "john.doe@sap.com",
      "type" : "work",
      "primary" : true
    }
  ]
}
```

```

] ,
"phoneNumbers" : [
  {
    "value" : "+123456789",
    "type" : "work"
  },
  {
    "value" : "+987654321",
    "type" : "mobile"
  }
],
"userType" : "Employee",
"preferredLanguage" : "E",
"timezone" : "CET",
"active" : true,
"groups" : [
  {
    "value" : "BUGR_ZCBUSERGROUP",
    "$ref" : "https://system.com.sap/bc/http/sap/aps_iam_api_scim/Groups/BUGR_ZCBUSERGROUP",
    "display" : "business user group",
    "type" : "userGroup"
  },
  {
    "value" : "BROL_BUSINESSROLE1",
    "$ref" : "https://system.com.sap/bc/http/sap/aps_iam_api_scim/Groups/BROL_BUSINESSROLE1",
    "display" : "business role 1",
    "type" : "authorization"
  },
  {
    "value" : "BROL_BUSINESSROLE2",
    "$ref" : "https://system.com.sap/bc/http/sap/aps_iam_api_scim/Groups/BROL_BUSINESSROLE2",
    "display" : "business role 2",
    "type" : "authorization"
  }
],
"urn:ietf:params:scim:schemas:extension:sap:2.0:User" : {
  "userUuid" : "GLOBAL_USER_ID"
},
"meta" : {
  "resourceType" : "User",
  "created" : "2015-05-22T12:53:56Z",
  "lastModified" : "2015-05-22T12:53:56Z",
  "version" : "20150522125356",
  "location" : "https://system.com.sap/bc/http/sap/aps_iam_api_scim/Users/CB0000000001"
}
}

```

### 4.2.9.5.16.3 Example: Group Endpoint

Please find an example of a group endpoint below:

#### ↔ Sample Code

```
{
  "schemas" : [
    "urn:ietf:params:scim:schemas:core:2.0:Group",
    "urn:ietf:params:scim:schemas:extension:sap:2.0:Group"
}
```

```

],
"id" : "BROL_BUSINESSROLE1",
"externalId" : "BUSINESSROLE1",
"displayName" : "business role 1",
"members" : [
  {
    "value" : "CB0000000001",
    "$ref" : "https://system.com/sap/bc/http/sap/aps_iam_api_scim/Users/
CB0000000001",
    "display" : "John Doe"
  }
],
"urn:ietf:params:scim:schemas:extension:sap:2.0:Group" : {
  "type" : "authorization",
  "supportedOperations" : "membership"
},
"meta" : {
  "resourceType" : "Group",
  "created" : "2018-12-11T19:18:05Z",
  "lastModified" : "2021-09-08T12:38:32Z",
  "version" : "20210908123832",
  "location" : "https://system.com/sap/bc/http/sap/aps_iam_api_scim/Groups/
BROL_BUSINESSROLE1"
}
}

```

## 4.2.9.5.17 Manage Git Repository

The Manage Git Repository API allows you to manage software components (Git repositories) on an SAP BTP ABAP Environment system.

### Overview

On an ABAP Environment system, software components, which represent Git repositories, can be controlled with Git operations. This includes cloning to an ABAP Environment system, pulling updates and working with branches (create, delete, checkout). These operations can be used with this API.

**Technical Name:** MANAGE\_GIT\_REPOSITORY

**OData Version:** 2.0

**Root URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY

### Permissions

In order to access this service, you need to:

- Create a *Communication User* as described in [How to Create Communication Users \[page 2594\]](#).
- Create a *Communication System* as described in [How to Create Communication Systems](#).
- Create a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 2596\]](#).
- Select the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.

## Service Structure

Service Metadata URI: sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/\$metadata

## Service Entities

Service Entity	Description
<a href="#">Branches [page 1157]</a>	Branches of a software component
<a href="#">Clones [page 1164]</a>	Clone job of a software component
<a href="#">Execution Log [page 1168]</a>	Execution Log of a software component
<a href="#">Pull [page 1171]</a>	Pull job of a software component
<a href="#">Transport Log [page 1175]</a>	Transport log of an imported transport

### 4.2.9.5.17.1 Branches

Resource path: /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Branches

## Operations

### CRUD Operations

HTTP Method	Operation
GET	<a href="#">Read Branches [page 1158]</a>
POST	<a href="#">Create Branch [page 1159]</a>
DELETE	<a href="#">Delete Branches [page 1161]</a>

### Custom Operations

HTTP Method	Operation Type	Operation	URI
POST	Action	<a href="#">Checkout Branches [page 1162]</a>	/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/checkout_branch

## Parameters

Property	Description
branch_name	name of the branch
sc_name	name of the software component
is_active	indicates if the branch is active in the system
critivality	

Property	Description
derived_from	name of the original branch from which the branch was derived
created_by	user who created the branch
created_on	date when the branch was created
commit_id	latest commit ID on the branch
commit_message	commit message of the latest commit
last_commit_by	user who committed the last commit
last_commit_on	date of the latest commit

## 4.2.9.5.17.1.1 Read Branches

Read one or multiple branches .

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Branches

**Operation Type:** CRUD

**HTTP Method:** GET

#### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

#### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
branch_name	yes	string	name of the branch	query string

#### Request Example

##### Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
Branches(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY' HTTP/1.1
```

```
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Branch was read successfully
404	Not found	Could not read branch; software component or branch may not exist..

### Request Payload Example

#### Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.BranchesType"
    },
    "branch_name": "newBranch",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "is_active": false,
    "criticality": 0,
    "derived_from": "main",
    "created_by": "CC0000000001",
    "created_on": "/Date(1584967657000+0000)/",
    "last_commit_on": "/Date(1584634658000+0000)/"
  }
}
```

## 4.2.9.5.17.1.2 Create Branch

Create a new branch.

## Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Branches

**Operation Type:** CRUD

**HTTP Method:** POST

### Request Headers

Header	Required	Values
Accept	no	application/json
		application/xml
Content-Type	no	application/json
		application/xml
x-csrf-token	yes	Value of x-csrf-token

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	Request body
branch_name	yes	string	name of the branch	Request body
derived_from	yes	string	name of the original branch from which the branch was derived	Request body

### Request Example

#### ↔ Sample Code

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
{
  "sc_name" : "/DMO/GIT_REPOSITORY",
  "derived_from" : "main",
  "branch_name" : "newBranch"
}
```

## Response

### Response Status and Error Codes

Code	Reason	Description
201	Created	Branch was created successfully
400	Bad Request	Could not create a branch due to the values passed in the request body.

### Request Payload Example

↔ Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Brances(branch_name='newBranch',sc_name='/DMO/GIT_REPOSITORY')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.BranchesType"
    },
    "branch_name": "newBranch",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "is_active": false,
    "criticality": 0,
    "derived_from": "master",
    "created_by": "CC0000000001",
    "created_on": "/Date(1584967657000+0000)/",
    "last_commit_on": "/Date(1584634658000+0000)/"
  }
}
```

### 4.2.9.5.17.1.3 Delete Branches

Delete a branch.

## Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Brances

**Operation Type:** CRUD

**HTTP Method:** DELETE

## Request Headers

## Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
branch_name	yes	string	name of the branch	query string

## Request Example

### ↔ Sample Code

```
DELETE /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches(branch_name='my-to-be-deleted-branch',sc_name='/DMO/GIT_REPOSITORY') HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

## Response Headers

### ↔ Sample Code

```
HTTP/1.1 204 No Content
```

## Response Status and Error Codes

Code	Reason	Description
204	No content	Branch was deleted successfully
400	Bad Request	Could not delete branch due to the values passed in the request body.

## 4.2.9.5.17.1.4 Checkout Branches

Checkout a branch.

## Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Branches

**Operation Type:** CRUD

**HTTP Method:** POST

## Request Headers

Header	Required	Values
Content-Type	no	application/json
		application/xml
Accept	no	application/json
		application/xml
x-csrf-token	yes	Value of x-csrf-token

## Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
branch_name	yes	string	name of the branch	query string

## Request Example

### Sample Code

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/checkout_branch?  
branch_name='newBranch'&sc_name='/DMO/GIT_REPOSITORY' HTTP/1.1  
Host: host.com  
Authentication: basicAuthentication  
X-csrf-token: xCsrfToken  
Content-Type: application/json  
Accept: application/json
```

## Response

### Response Status and Error Codes

Code	Reason	Description
200	OK	Branch was checked out successfully
400	Bad Request	Could not check out a branch due to the values passed in the request body

### Response Payload Example

#### Sample Code

```
{  
  "d": {  
    "__metadata": {
```

```

        "id": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
Pull(guid'UUID')",
        "uri": "https://host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
Pull(guid'UUID')",
        "type": "cds_sd_a4c_a2g_gha_sc_web_api.PullType"
    },
    "uuid": "UUID",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "namespace": "",
    "status": "R",
    "status_descr": "Running",
    "start_time": "/Date(1571227437000+0000)/",
    "change_time": "/Date(1571227472000+0000)/",
    "criticality": 2,
    "user_name": "CC0000000001",
    "to_Execution_log": {
        "__deferred": {
            "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log"
        }
    },
    "to_Transport_log": {
        "__deferred": {
            "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log"
        }
    }
}
}

```

## 4.2.9.5.17.2 Clones

**Resource path:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Clones

### Operations

#### CRUD Operations

HTTP Method	Description
GET	<a href="#">Read Clones [page 1165]</a>
POST	<a href="#">Create Clone [page 1167]</a>

#### Parameters

Property	Description
uuid	ID of the job
sc_name	Name of the software component
branch_name	Name of the branch

Property	Description
commit_id	Commit ID
import_type	Type of import
status	Status of the job
status_descr	Status description
user_name	Name of the user who triggered the action
start_time	Start time of the job
change_time	Last update of the job

## 4.2.9.5.17.2.1 Read Clones

Read a clone job. The entity is only available after the job has finished. To track the status of the job, you can use the GET /Pull(guid'UUID') request.

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Clones

**Operation Type:** CRUD

**HTTP Method:** GET

### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
branch_name	yes	string	name of the branch	query string
uuid	yes	string	ID of the job	query string

## Request Example

### ↳ Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4',branch_name='main',sc_name='/DMO/GIT_REPOSITORY') HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Read clone successfully
400	Bad Request	Could not read the clone entity. Please check the parameters.

### Request Payload Example

### ↳ Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://4c7c0f11-da7c-43cf-b60c-e6e308144458.abap.stagingaws.hanavlab.ondemand.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4',sc_name='/%2FDMO%2FGIT_REPOSITORY,branch_name='main'))",
      "uri": "https://4c7c0f11-da7c-43cf-b60c-e6e308144458.abap.stagingaws.hanavlab.ondemand.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4',sc_name='/%2FDMO%2FGIT_REPOSITORY,branch_name='main'))",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.ClonesType"
    },
    "uuid": "02409ac8-dc6a-1edb-908e-31e4b44596d4",
    "sc_name": "/DMO/GIT_REPOSITORY",
    "branch_name": "main",
    "commit_id": "",
    "import_type": "Clone",
    "namespace": "",
    "status": "S",
    "status_descr": "Success",
    "criticality": 3,
    "user_name": "CC0000000001",
    "start_time": "/Date(1608214205000+0000)/"
```

```
        "change_time": "/Date(1608214243000+0000)/"  
    }  
}
```

## 4.2.9.5.17.2.2 Create Clone

Create a clone job.

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Clones

**Operation Type:** CRUD

**HTTP Method:** POST

#### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
Content-Type	no	application/json application/xml
x-csrf-token	no	fetch

#### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	Request body
branch_name	no	string	name of the branch	Request body
commit_id	no	string	commit ID	Request body

#### Request Example

##### Sample Code

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones HTTP/1.1  
Host: host.com  
Authentication: basicAuthentication  
X-csrf-token: xCsrfToken  
Content-Type: application/json  
Accept: application/json
```

```
{  
    "sc_name" : "/DMO/GIT_REPOSITORY"  
    "branch_name": "main"  
}
```

## Response

### Response Status and Error Codes

Code	Reason	Description
201	Created	Created a clone job
400	Bad Request	Could not read the clone job. Please check the parameters.

### Request Payload Example

#### ↔ Sample Code

```
{  
    "d": {  
        "  
        " __metadata": {  
            "id":  
                "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid=guid'02ceab7d-ff04-1eda-b1ea-eb7fdbf28bc4',sc_name='ZHM_INITIAL_TEST',branch_name='')",  
            "uri":  
                "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid=guid'02ceab7d-ff04-1eda-b1ea-eb7fdbf28bc4',sc_name='ZHM_INITIAL_TEST',branch_name='')",  
            "type": "cds_sd_a4c_a2g_gha_sc_web_api.ClonesType"  
        },  
        "uuid": "02ceab7d-ff04-1eda-b1ea-eb7fdbf28bc4",  
        "sc_name": "/DMO/GIT_REPOSITORY",  
        "branch_name": "",  
        "import_type": "Clone",  
        "namespace": "",  
        "status": "S",  
        "status_descr": "Success",  
        "criticality": 3,  
        "user_name": "administrator@example.com",  
        "start_time": "/Date(1594898863000+0000)/",  
        "change_time": "/Date(1594898891000+0000)/"  
    }  
}
```

## 4.2.9.5.17.3 Execution Log

**Resource path:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/ExecutionLogs

## Operations

### CRUD Operations

HTTP Method	Description
GET	<a href="#">Read Execution Log [page 1169]</a>

### Parameters

Property	Description
uuid	ID of the job
index_no	Index number of the log entry
type	Type of the log (information, warning, etc...)
descry	Description text of this log entry
timestamp	Timestamp of the log entry

### 4.2.9.5.17.3.1 Read Execution Log

Read the execution log of a specific job.

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/to\_Execution\_log

**Operation Type:** CRUD

**HTTP Method:** GET

### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
uuid	yes	string	ID of the job	query string

## Request Example

### ↳ Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4')/to_Execution_log HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Read execution log successfully
400	Bad Request	Could not find an execution log with the specified UUID.

### Request Payload Example

### ↳ Sample Code

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')",
          "uri": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')"
        },
        "index_no": "1",
        "uuid": "UUID",
        "type": "Information",
        "descr": "Step X: MESSAGE",
        "timestamp": "/Date(1571227438000+0000)/",
        "criticality": 0
      }
    ]
  }
}
```

## 4.2.9.5.17.4 Pull

**Resource path:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Pull

### Operations

#### CRUD Operations

HTTP Method	Description	Operation
GET	Read pull	
POST	Trigger a pull job	

#### Custom Operations

HTTP Method	Description	Operation	URI
GET	Associationl		/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log
GET	Associationl		/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log

### Parameters

Property	Description
uuid	ID of the job
sc_name	Name of the software component
branch_name	Name of the branch
commit_id	Commit ID
import_type	Type of import
status	Status of the job
status_descr	Status description
user_name	Name of the user who triggered the action
start_time	Start time of the job
change_time	Last update of the job

## 4.2.9.5.17.4.1 Read Pull

Read a pull job.

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Pull

**Operation Type:** CRUD

**HTTP Method:** GET

#### Request Headers

Header	Required	Values
Accept	no	application/json
		application/xml
x-csrf-token	no	fetch

#### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
uuid	yes	string	ID of the job	query string

#### Request Example

↔ Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4') HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

### Response

#### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

## Response Status and Error Codes

Code	Reason	Description
200	OK	Read pull successfully
400	Bad Request	Could not find a pull entity with the specified UUID.

## Request Payload Example

↔ Sample Code

```
{  
    "d": {  
        "__metadata": {  
            "id": "https://host.com/sap/opu/odata/sap/  
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')", "uri": "https://host.com/sap/opu/  
odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')", "type":  
"cds_sd_a4c_a2g_gha_sc_web_api.PullType"  
        }  
        ,  
        "uuid": "UUID ",  
        "sc_name": "/DMO/GIT_REPOSITORY ",  
        "namespace": "",  
        "status": "R",  
        "status_descr": "Running",  
        "start_time": "/Date(1571227437000+0000)/",  
        "change_time": "/Date(1571227472000+0000)/",  
        "criticality": 2,  
        "user_name": "CC0000000001",  
        "to_Execution_log": {  
            "__deferred": {  
                "uri": "https://host.com/sap/opu/odata/sap/  
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log"  
            }  
        }  
        ,  
        "to_Transport_log": {  
            "__deferred": {  
                "uri": "https://host.com/sap/opu/odata/sap/  
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log"  
            }  
        }  
    }  
}
```

## 4.2.9.5.17.4.2 Create Pull

Create a pull job.

### Request

URI: /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Pull

**Operation Type:** CRUD

**HTTP Method:** POST

## Request Headers

Header	Required	Values
Accept	no	application/json
		application/xml
Content-Type	no	application/json
		application/xml
x-csrf-token	yes	Value of x-csrf-token

## Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	Request body
commit_id	no	string	commit ID	Request body

## Request Example

### Sample Code

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
{
    "sc_name" : "/DMO/GIT_REPOSITORY"
}
```

## Response

### Response Status and Error Codes

Code	Reason	Description
201	Created	Created a pull job
400	Bad Request	Could not read the pull job. Please check the parameters.

## Request Payload Example

### ↳ Sample Code

```
{
  "d": {
    "__metadata": {
      "id": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')",
      "uri": "https://host.com/sap/opu/
odata/sap/MANAGE_GIT_REPOSITORY/Pull(guid'UUID')",
      "type": "cds_sd_a4c_a2g_gha_sc_web_api.PullType"
    }
  },
  "uuid": "UUID",
  "sc_name": "/DMO/GIT_REPOSITORY",
  "namespace": "",
  "status": "R",
  "status_descr": "Running",
  "start_time": "/Date(1571227437000+0000)/",
  "change_time": "/Date(1571227472000+0000)/",
  "criticality": 2,
  "user_name": "CC0000000001",
  "to_Execution_log": {
    "__deferred": {
      "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Execution_log"
    }
  },
  "to_Transport_log": {
    "__deferred": {
      "uri": "https://host.com/sap/opu/odata/sap/
MANAGE_GIT_REPOSITORY/Pull(guid'UUID')/to_Transport_log"
    }
  }
}
```

## 4.2.9.5.17.5 Transport Log

**Resource path:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/TransportLogs

## Operations

### CRUD Operations

HTTP Method	Operation
GET	<a href="#">Read Transport Log [page 1176]</a>

## Parameters

Property	Description
uuid	ID of the job
index_no	Index number of the log entry
type	Type of the log (information, warning, etc...)
descry	Description text of this log entry
timestamp	Timestamp of the log entry

## 4.2.9.5.17.5.1 Read Transport Log

### Request

URI: /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/to\_Transport\_log

Operation Type: CRUD

HTTP Method: GET

### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
uuid	yes	string	ID of the job	query string

### Request Example

#### Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Pull(uuid='02409ac8-dc6a-1edb-908e-31e4b44596d4')/to_Transport_log HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Read transport log successfully
400	Bad Request	Could not find a transport log with the specified UUID.

### Request Payload Example

↔ Sample Code

```
{ {
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')",
          "uri": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/ExecutionLogs(index_no=1m,uuid=guid'UUID')",
          "type": "cds_sd_a4c_a2g_gha_sc_web_api.ExecutionLogsType"
        }
      },
      {
        "index_no": "1",
        "uuid": "UUID",
        "type": "Information",
        "descr": "Step X: Message",
        "timestamp": "/Date(1571227438000+0000)/",
        "criticality": 0
      }
    ]
  }
}
```

## 4.2.9.5.17.6 Tags

Resource path: /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Tags

## Operations

### CRUD Operations

HTTP Method	Operation
GET	<a href="#">Read Entities from Tag [page 1178]</a>
POST	<a href="#">Create Tag [page 1179]</a>
DELETE	<a href="#">Delete Tag [page 1182]</a>

### Custom Operations

HTTP Method	Operation Type	Operation	URI
POST	Action	<a href="#">Read Tags by Key [page 1181]</a>	/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Tags(sc_name='{sc_name}',commit_id='{commit_id}',tag_name='{tag_name}')

## 4.2.9.5.17.6.1 Read Entities from Tag

Read entities from tags.

### Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Tags

**Operation Type:** CRUD

**HTTP Method:** GET

### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

### Request Parameters

No parameters.

## Request Example

### ↔ Sample Code

```
GET /sap/opu/odata/sap/Tags  
Host: host.com  
Authentication: basicAuthentication  
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Branch was read successfully
404	Not found	Could not read branch; software component or branch may not exist..

### Request Payload Example

### ↔ Sample Code

## 4.2.9.5.17.6.2 Create Tag

Create a new tag.

## Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Tags

**Operation Type:** CRUD

**HTTP Method:** POST

## Request Headers

Header	Required	Values
Accept	no	application/json
		application/xml
Content-Type	no	application/json
		application/xml
x-csrf-token	yes	Value of x-csrf-token

## Request Parameters

No parameters.

## Request Body

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	Request body
commit_id	yes	string	long commit id	Request body
tag_name	yes	string	name of the tag	Request body

### Sample Code

```
POST/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Tags
HTTP/1.1
Host: host.com
Authentication: basicAuthentication
X-csrf-token: xCsrfToken
Content-Type: application/json
Accept: application/json
{
  "sc_name": "string",
  "commit_id": "string",
  "tag_name": "string",
  "tag_description": "string",
  "tag_created_by": "string",
  "tag_created_on": "/Date(1492098664000)/",
  "short_commit_id": "string",
  "message": "string",
  "author": "string",
  "author_mail": "string"
}
```

## Response

### Response Status and Error Codes

Code	Reason	Description
201	Created	Tag was created successfully
400	Bad Request	Could not create a tag due to the values passed in the request body.

### Request Payload Example

↔ Sample Code

## 4.2.9.5.17.6.3 Read Tags by Key

Read entities from tags.

### Request

**URI:** : /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/Tags(sc\_name='{sc\_name}',commit\_id='{commit\_id}',tag\_name='{tag\_name}')

**Operation Type:** CRUD

**HTTP Method:** GET

### Request Headers

Header	Required	Values
Accept	no	application/json application/xml
x-csrf-token	no	fetch

### Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
commit_id	yes	string	long commit id	query string

Parameter	Required	Data Type	Description	Parameter Type
tag_name	yes	string	name of the tag	query string

## Request Example

### ↔ Sample Code

```
GET /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/
Tags(sc_name='{sc_name}',commit_id='{commit_id}',tag_name='{tag_name}')
HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

Header	Description
x-csrf-token	Token, which can be used for POST requests

### Response Status and Error Codes

Code	Reason	Description
200	OK	Tag was read successfully
404	Not found	Could not read tag; software component or commit id may not exist..

### Request Payload Example

### ↔ Sample Code

## 4.2.9.5.17.6.4 Delete Tag

Delete a branch.

## Request

**URI:** /sap/opu/odata/sap/MANAGE\_GIT\_REPOSITORY/  
 Tags(sc\_name='{sc\_name}',commit\_id='{commit\_id}',tag\_name='{tag\_name}')

**Operation Type:** CRUD

**HTTP Method:** DELETE

## Request Headers

Header	Required	Values
Accept	no	application/json
		application/xml
x-csrf-token	yes	Value of x-csrf-token

## Request Body

Parameter	Required	Data Type	Description	Parameter Type
sc_name	yes	string	name of the software component	query string
commid_id	yes	string	long commit id	query string
tag_name	yes	string	name of the branch	query string

## Request Example

### ↔ Sample Code

```
DELETE /sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Branches(branch_name='my-to-be-deleted-branch',sc_name='/DMO/GIT_REPOSITORY') HTTP/1.1
Host: host.com
Authentication: basicAuthentication
Accept: application/json
```

## Response

### Response Headers

### ↔ Sample Code

```
HTTP/1.1 204 No Content
```

### Response Status and Error Codes

Code	Reason	Description
204	No content	Tag was deleted successfully
400	Bad Request	Could not delete tag due to the values passed in the request body.

## 4.2.9.6 Integrating On-Premise Systems

Set up the Cloud Connector to enable communication from the ABAP environment to your on-premise systems using Remote Function Calls (RFC) and HTTP calls.

### Concept

For each subaccount, both a default Connectivity and Destination service instance are set up automatically in the SAP provider account. Using these default instances, you only need to configure the required destinations on subaccount level to enable communication to your on-premise systems.

In this default scenario, you must connect the Cloud Connector to and configure trust for your Cloud Foundry subaccount as described in [SAP BTP Connectivity: Cloud Connector](#).

#### ⚠ Restriction

This scenario requires a Cloud Connector as of version 2.12.3.

The host name of the Cloud Connector is not needed because the Cloud Connector itself connects **to the Cloud**, but it is **never connected from the Cloud**.

#### ⓘ Note

For more information on the deprecated scenario that uses the Connectivity service in the Neo environment, see [Create a Communication Arrangement for Cloud Connector Integration \(Deprecated\)](#) [page 1185].

After you have completed the setup, a connection from the ABAP environment tenant to an on-premise system is established in the following order:

1. The ABAP environment tenant fetches the destination from the Destination service instance.
2. The ABAP environment tenant requests to open the tunnel connection through the Connectivity service.
3. The Connectivity service tells the Cloud Connector to open the connection to this specific ABAP environment tenant using the admin connection.
4. The Cloud Connector opens a tunnel connection to the ABAP environment tenant using its public tenant URL.
5. After the tunnel is established, it can be used for actual data connection using RFC and HTTP(S) protocols.

### Next Steps

To set up your actual data connections between your ABAP environment and on-premise systems, you must configure the required destinations. See [Enable On-Premise Connectivity](#) [page 1187].

## Troubleshooting

In case of issues using the Cloud Connector, see [Cloud Connector Troubleshooting](#) for a troubleshooting guide.

### 4.2.9.6.1 Create a Communication Arrangement for Cloud Connector Integration (Deprecated)

Learn how to create a communication arrangement for communication scenario SAP\_COM\_0200 to integrate the Cloud Connector.

#### Prerequisites

##### ⓘ Note

This procedure is deprecated and is only available for systems provisioned before May 2022. It was replaced with a scenario that does not require a Neo subaccount. To learn about the steps to migrate to a Cloud Foundry-only usage, see section **Next Steps** below.

To set up your ABAP environment, you have to create a communication arrangement for communication scenario SAP\_COM\_0200 – Cloud Connector Integration, that requires the following:

- An administrative user in the ABAP environment tenant.
- Increased quota for the ABAP runtime.
- An ABAP service instance set up in Cloud Foundry.
- Your Neo subaccount name.
- The name of the region host of your Neo subaccount

##### ⓘ Note

Please check out the supported Neo subaccounts for on-premise connectivity of the ABAP environment. See SAP Note [2765161](#).

- Installation of Cloud Connector version 2.11 or higher. See [Cloud Connector: Installation](#)
- Initial configuration for Cloud Connector and the Neo subaccount. See [Cloud Connector: Initial Configuration](#)

##### ⓘ Note

The Cloud Connector is connected to the Neo subaccount and is displayed in the Cloud Cockpit, section [Connectivity](#) under [Cloud Connectors](#).

- User credentials for the Cloud Connector admin role in the Neo subaccount

## Procedure

1. To create a communication system that represents your Neo subaccount, open the *Communication Systems* app from the SAP Fiori launchpad in your Cloud Foundry subaccount and select *New*.
2. Provide a meaningful system name and choose *Create* to proceed.

### Note

You have to disable the destination service first by using the *on/off* slider button.

Enter the following information:

Field	Action
<i>Host Name</i>	Enter the name of the region host of your Neo subaccount.  If your Neo URL is <a href="https://account.xyz.hana.on-demand.com">https://account.xyz.hana.on-demand.com</a> , you need to enter <b>xyz.hana.ondemand.com</b> , where <b>xyz</b> represents the region name. See <a href="#">Regions and Hosts Available for the Neo Environment</a> .
<i>HTTPS Port</i>	Enter <b>443</b> .
<i>Checkbox Use Cloud Connector</i>	Make sure this checkbox is <b>not checked</b> .
User for Outbound Communication	a) Scroll down to this section.  b) Add a new user.  c) In the dropdown list for <i>Authentication Method</i> , select a user name and password.  d) Enter the user name and password of the Cloud Connector admin user of the Neo subaccount.

For more information on how to create a communication system, see [How to Create Communication Systems](#).

3. Save the communication system to activate it.
4. To create a communication arrangement that activates the connection to the Neo subaccount, open the *Maintain Communication Arrangements* app from the SAP Fiori Launchpad and click *New*.
5. Select scenario **SAP\_COM\_0200** and enter a meaningful arrangement name.
6. Click *Create* to proceed.

Enter the following information:

Field	Action
<i>Communication System</i>	Enter the name of the communication system created in steps 1-3, or select it from the value help.

Field	Action
▶ <a href="#">Additional Properties</a> ▶ <a href="#">Account Name</a> ▶	Enter the name of your Neo subaccount.
▶ <a href="#">Outbound Communication</a> ▶ <a href="#">User Name</a> ▶	

For more information on how to create a communication arrangement, see [How to Create a Communication Arrangement \[page 2596\]](#).

7. Save the communication arrangement to activate it.
8. Carry out the initial configuration for Cloud Connector and the Neo subaccount to make it operational for connections between your Neo subaccount and the on-premise systems.

## Results

You have associated the ABAP environment tenant with the Neo subaccount. This enables the ABAP environment tenant to request the Neo environment to open a tunnel connection.

## Next Steps

After completing the setup of communication scenario SAP\_COM\_0200, you are ready to set up your actual data connections between your ABAP environment and on-premise systems. This requires the configuration of other communication systems and communication arrangements in the ABAP environment tenant as well as the configuration of Cloud Connector. See [Enable On-Premise Connectivity \[page 1187\]](#).

To **migrate** to the usage of the **Cloud Foundry Connectivity service**, proceed as follows:

1. Configure trust in the Cloud Connector for the Cloud Foundry subaccount in which the ABAP instance resides.
2. Delete communication arrangement SAP\_COM\_0200.
3. Destinations do not need to be modified if the same LocationID is used.

## 4.2.9.6.2 Enable On-Premise Connectivity

Create an HTTP and an RFC destination to enable communication from the ABAP environment to your on-premise systems.

### Prerequisites

- You have installed Cloud Connector version 2.12.3 or higher, see [Cloud Connector: Installation](#).

- You have done the initial configuration for the Cloud Connector, see [Cloud Connector: Initial Configuration](#).
- If you want to use principal propagation as authentication type, the Cloud Connector must be configured to support this authentication type. See [Cloud Connector: Configuring Principal Propagation](#).  
Also, make sure that no communication arrangement to the deprecated communication scenario SAP\_COM\_0200 exists.
- If you use more than one Cloud Connector in your subaccount, you have assigned a location ID to each of these Cloud Connectors. See [Managing Subaccounts](#) (section **Procedure**, step 4).

## Procedure

To enable on-premise connectivity, you must set up or reuse an HTTP destination or an RFC destination with **<Proxy Type> OnPremise**:

- [Set Up HTTP Communication \[page 1003\]](#)
- [Service Consumption Model as RFC Consumer \[page 1006\]](#)

### 4.2.9.7 Integrating the ABAP Environment with SAP S/4HANA Cloud

You can integrate the SAP BTP, ABAP environment with an SAP S/4HANA Cloud Public Edition system. This allows ABAP developers to implement outbound service calls from the ABAP environment to the SAP S/4HANA Cloud Public Edition system.

## Context

Developers in SAP BTP, ABAP environment would like to access data from business services in SAP S/4HANA Cloud Public Edition as part of their applications.

This documentation provides you with a high-level description of how to integrate the two systems and how to implement service calls from SAP BTP, ABAP environment to SAP S/4HANA Cloud Public Edition. This is complemented by a tutorial group that provides a detailed, step-by-step description based on an example.

## Overview of Implementation Activities

To set up remote connectivity it is necessary to perform some steps in both involved systems, as the connection must be enabled by both communication partners.

In the scenario that we are considering, an SAP S/4HANA Cloud Public Edition service is called from SAP BTP, ABAP environment. In other words, SAP S/4HANA Cloud Public Edition is the *inbound* communication partner, while SAP BTP, ABAP environment is the *outbound* communication partner.

Following, you can find an overview of necessary steps, grouped by the executing persona.

### **Developer in SAP BTP, ABAP environment**

- *Create a communication scenario*

As a developer you create an outbound service that represents the service of an SAP S/4HANA Cloud Public Edition system. You add this outbound service to a communication scenario and maintain the supported authentication method.

- *Create a service consumption model*

As a developer you can use the metadata of a given service to create a service consumption model. Although not strictly necessary, this greatly simplifies the consumption of the remote service from your ABAP code.

- *Implement the application*

The actual service call itself needs to be implemented. This can be part of a frontend application, API service, application job, console application, etc. The communication artifacts and the service consumption model are reused to create compact and understandable code.

### **Administrator in SAP S/4HANA Cloud Public Edition**

- *Expose the service*

As administrator you need to make the required service available for remote access. To do so, you create a communication arrangement for a scenario containing the API as an inbound service. Along with the arrangement, you define a communication system and an inbound communication user that shall be used to establish the connection.

#### *Configure the outbound communication*

As administrator you create a communication arrangement based on the scenario implemented by the developer. The arrangement is tied to a communication system that identifies the SAP S/4HANA Cloud Public Edition system as the target host.

Tutorial group [Integrate the SAP BTP, ABAP environment with SAP S/4HANA Cloud, public edition](#) shows how to consume the business partner OData API service in SAP S/4HANA Cloud Public Edition from a console application in SAP BTP, ABAP environment using basic authentication (communication user) and OAuth 2.0 SAML Bearer Assertion (propagation of business user).

## **4.2.9.8 Developing Metrics for Health Monitoring**

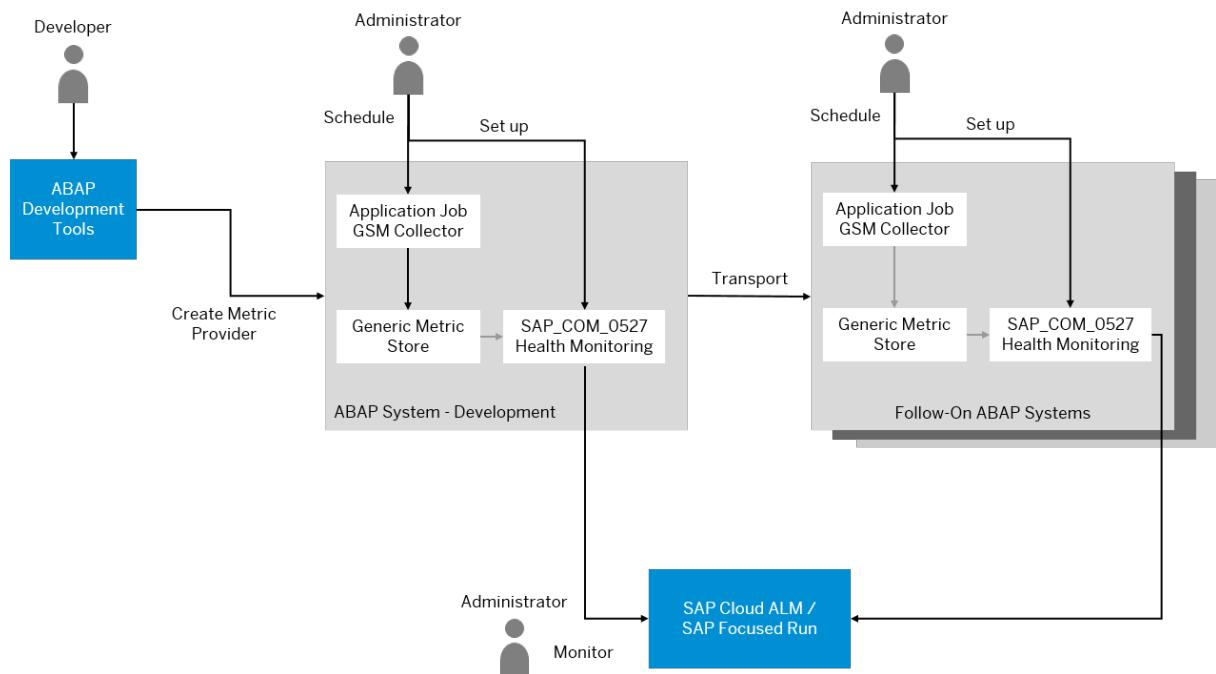
Learn how you can add your own metrics to Health Monitoring in SAP Cloud ALM or SAP Focused Run.

### **Use Case**

SAP Cloud ALM provides you with insights regarding the health of Cloud services. In the overview dashboard, all health monitoring metrics are exposed, and for each metric, a detailed history is available. While a number of metrics defined by SAP are available for checking customer-initiated operational issues, you might want to add your own application-specific metrics. These metrics serve to monitor the health of your own customer

or partner solutions running on the ABAP environment. Similarly, you might want to have your own metrics in Health Monitoring of SAP Focused Run.

## Architecture Overview



The central object in this context is the metric provider, an ABAP repository object that references an implementing class. The class defines a metric model for your application and implements the metric measurement. The metric model defines, for example, what kind of metrics are available in the system, how often a metric is measured, its tags, or its unit of measurement.

An application job (the generic simple metric collector) runs regularly. It calls up the `GET_METRIC_VALUES()` method of all metric providers and stores all measured values as metrics in a generic format in the metric store. After an administrator has set up a communication arrangement based on the communication scenario `SAP_COM_0527`, the metrics are automatically pushed from the generic metric store to SAP Cloud ALM and SAP Focused Run.

## About This Documentation

In this documentation, you get information about creating your own metric providers as a developer. In addition, there's also a short information for administrators about creating an application job that calls up all created metric providers and transfers their measured values to the generic metric store.

For more information about setting up SAP Cloud ALM and SAP Focused Run, see [Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#).

## 4.2.9.8.1 Creating a Class for Metric Providers

To create a metric model for your customer or partner application, create a class in ABAP Development Tools (ADT) that implements the interface `IF_GSM_API_PROVIDER`.

### Prerequisites

You have a developer role for creating objects with ABAP for Cloud Development.

You have created an ABAP Cloud project and a package in ABAP Development Tools.

### Context

With the class, you define the metric model and measurement.

#### ⓘ Note

For code samples, see <https://github.com/SAP-samples/abap-platform-ops> .

### Procedure

1. In your ABAP Cloud project and package, create a class implementing the interface `IF_GSM_API_PROVIDER`.

As a result, you get a class with the following methods: `INITIALIZE( )`, `DEFINE_MODEL( )`, and `GET_METRIC_VALUES( )`.

2. When you edit this class, we recommend that you define constants for the metric names and values that are exactly the same for methods `DEFINE_MODEL( )` and `GET_METRIC_VALUES( )`. All metric names that do not match are ignored at runtime.

For example, your code might look as follows:

#### ↔ Sample Code

```
CONSTANTS:  
    BEGIN OF e_metric_group_id,  
        demo_group TYPE if_gsm_api_types=>tv_metric_group_id VALUE  
        'zdemo_group_01',  
        END OF e_metric_group_id .  
CONSTANTS:  
    BEGIN OF e_metric_id,  
        demo_metric_id TYPE if_gsm_api_types=>tv_metric_id VALUE  
        'zdemo_metric_01',  
        END OF e_metric_id .  
CONSTANTS:  
    BEGIN OF e_attribute_id,
```

```
demo_attr_01 TYPE if_gsm_api_types=>tv_attribute_id VALUE  
'zdemo_attr_01',  
END OF e_attribute_id .
```

You create one metric group and assign your metrics to the group. The metric group is just for semantical grouping and holder for meta data. It has no physical impact on the metric store or the data collection for metrics later.

3. Redefine the methods `DEFINE_MODEL()` and `GET_METRIC_VALUES()` according to your business needs.

## Related Information

[Creating a Metric Model \[page 1192\]](#)

[Defining the Metric Measurement \[page 1193\]](#)

### 4.2.9.8.1.1 Creating a Metric Model

In the `DEFINE_MODEL()` method, you specify your metric model, including the metric IDs, their data types, or their units of measurement.

#### Prerequisites

You've created a class for metric providers (see [Creating a Class for Metric Providers \[page 1191\]](#)).

#### Procedure

In the class that you've created, change the `DEFINE_MODEL()` method according to your business needs.

For example, your method might be as follows:

#### Sample Code

```
METHOD if_gsm_api_provider~define_model.  
DATA: lo_api_metric_group TYPE REF TO if_gsm_api_metric_group.  
lo_api_metric_group = io_model->add_metric_group(  
    iv_metric_group_id      = me->e_metric_group_id-demo_group  
    iv_metric_group_name    = `Demo Client API Provider` ##NO_TEXT  
    iv_category            = if_gsm_api_constants=>e_category-performance  
    iv_severity             = if_gsm_api_constants=>e_severity-_2_low ).  
lo_api_metric_group->add_target(  
    iv_target_type          = if_gsm_api_constants=>e_target_type-health ).  
lo_api_metric_group->add_metric(  
    iv_metric_id            = me->e_metric_id-demo_metric_id  
    iv_metric_name          = `Demo Client API Metric` ##NO_TEXT
```

```

    iv_unit           = if_gsm_api_constants=>e_unit-seconds
    iv_data_type     = if_gsm_api_constants=>e_data_type-integer
    iv_period        = if_gsm_api_constants=>e_period-every_05_minutes
    iv_description   = `Just a demo` ##NO_TEXT
    iv_metric_type   = if_gsm_api_constants=>e_metric_type-counter )-
>add_target(
    iv_target_type  = if_gsm_api_constants=>e_target_type-health .
ENDMETHOD.
```

Notes: For all input parameters, you can find the relevant enumerations in the released interface `IF_GSM_API_CONSTANTS`. Note, however, that there are a few additional considerations:

- The category of the metric group expresses the semantic nature of the metric, for example, whether the availability or performance of the system is affected.
- The attributes of a metric refer to the consumption of the metric:
  - **Period:** The period specifies the frequency with which the metric measurement is triggered. It implicitly derives a runtime quota. Metric providers that are executed more often must return the metrics faster. For example, metric providers running each minute must return values within 5 seconds. Otherwise, they will be banned by the framework in the future.  
The `period` attribute is the only control attribute. All other attributes don't have any runtime impacts.
  - **Metric type:** Metrics can be collected in different ways. Some metrics are of type `Counter`. A counter always goes up and is only reset at special events like system restart. For example, the number of processed requests is typically expressed as a counter. In contrast to a counter, a gauge metric can go up or down. The temperature or current memory usage are common examples of gauge metrics.
  - **Target type:** The target type is used to control the metric flow for different usage scenarios. Only the available target type `HEALTH` is supported. This target type registers the metric to be exported to Health Monitoring in SAP Cloud ALM and SAP Focused Run.
  - **Severity:** The severity is available, but not pushed to Health Monitoring.

## 4.2.9.8.1.2 Defining the Metric Measurement

In the `GET_METRIC_VALUES( )` method, you specify how the metrics are measured, including the requested metrics and the time frame for processing.

### Prerequisites

You've created a class for metric providers (see [Creating a Class for Metric Providers \[page 1191\]](#)).

### Context

At runtime, the generic metric provider framework calls the method `GET_METRIC_VALUES( )`. The method expects multiple input parameters like the requested metrics and the time frame for processing. If you include multiple metrics with different collection frequencies, the framework calls this method accordingly with the

corresponding metric IDs for which it requests to return data. When you provide the correct input in the GET\_METRIC\_VALUES( ) implementation, you avoid wasting work for metrics not being requested.

The measurement should be fast. If a measurement takes too much time, it blocks all the other measurements. Therefore, the metric provider framework automatically monitors the runtime of each metric provider. If a metric provider exceeds the runtime quota, the framework will ban the metric provider so that all the other metric providers can be collected.

## Procedure

In the class that you've created, change the GET\_METRIC\_VALUES( ) method according to your business needs.

For example, your method might be as follows:

### Sample Code

```
METHOD if_gsm_api_provider~get_metric_values.
  DATA: ls_metric_value      TYPE if_gsm_api_types=>ts_metric_value.
  DATA: lv_timestamp         TYPE timestamp.
  GET TIME STAMP FIELD lv_timestamp.
  ls_metric_value-metric_id = me->e_metric_id-demo_metric_id.
  IF iv_timestamp_start IS SUPPLIED AND iv_timestamp_start <> 0.
    ls_metric_value-value_timestamp = iv_timestamp_start.
  ELSEIF iv_timestamp_end IS SUPPLIED AND iv_timestamp_end <> 0.
    ls_metric_value-value_timestamp = iv_timestamp_end.
  ELSE.
    ls_metric_value-value_timestamp = lv_timestamp.
  ENDIF.
  ls_metric_value-value_count = 1.
  ls_metric_value-value_text = ''.
  INSERT VALUE #(
    attribute_id = me->e_attribute_id-demo_attr_01
    value        = 'None' ##NO_TEXT
  ) INTO TABLE ls_metric_value-attributes
    REFERENCE INTO DATA(lsr_mp_attr_01).
  " metric data
  "
  -----
  " Retrieval of the metric values at this point in time.
  DATA(lt_metrics) = me->mo_api_access->get_metrics( ).
  " transformation
  "
  -----
  " Transform the metric values format to the GSM data structures.
  LOOP AT lt_metrics REFERENCE INTO DATA(lrs_metric).
    lsr_mp_attr_01->value      = lrs_metric->attribute.
    ls_metric_value-value_sum = lrs_metric->value.
    ls_metric_value-value_max = ls_metric_value-value_sum.
    ls_metric_value-value_min = ls_metric_value-value_sum.
    IF ls_metric_value-value_sum > 10.
      ls_metric_value-value_rating = if_gsm_api_constants=>e_rating-
      _2_warning.
    ELSE.
      ls_metric_value-value_rating = if_gsm_api_constants=>e_rating-_1_ok.
    ENDIF.
    INSERT ls_metric_value INTO TABLE et_metric_values.
  ENDLOOP.
ENDMETHOD.
```

## 4.2.9.8.2 Creating a Metric Provider

You can create a metric provider in ABAP Development Tools to make your own metrics available for health monitoring of your ABAP system.

### Prerequisites

You've created a class implementing the interface `IF_GSM_API_PROVIDER`. This class defines your metric model.

### Context

With the creation of a metric provider, you register your metrics that are defined in the class for a periodic collection of values.

#### ⓘ Note

For more information about creating metric providers, see the documentation in ABAP Development Tools.

### Procedure

1. To create a metric provider in ABAP Development Tools, choose .
2. In the wizard, enter a package, name, and description.  
A metric provider is created.
3. In the metric provider, as implementing object, enter the class that defines your metric model for this metric provider (see [Creating a Class for Metric Providers \[page 1191\]](#) ).
4. Under *Execution*, select the priority with which the values of this metric provider must be collected.

#### ⓘ Note

The execution mode is set to *Job*, which you can't change. The collection of metric values is executed by an application job that your administrator must schedule (see [Collect Metric Provider Values \(Administrator\) \[page 1197\]](#)). The application job runs regularly and collects the values of all metric providers using their `GET_METRIC_VALUES()` methods.

Set a higher priority for metric providers that you consider as more important than others. A higher priority ensures that the values for these metric providers are collected first by the application job. Make sure that metric providers that are dependent on the values of other metric providers have a lower priority than the metric providers on which they depend.

5. Leave the field *Scope Dependent* set to *Yes*.

### 4.2.9.8.3 Testing the Metric Providers

You can use class `CL_GSM_API_TEST` to create unit tests for your metric providers.

#### Context

With unit tests, you can simulate the API that retrieves the lower-level raw data. You focus here on the transformation logic to the structures of the generic metric store.

#### Procedure

1. In ABAP Development Tools (ADT), you can find the class `CL_GSM_API_TEST` as released object in the package `S_GSM_API`.
2. To create unit tests, implement the methods of the class `CL_GSM_API_TEST`.

### 4.2.9.8.4 Considerations for Metric Provider Updates

When you update an existing metric provider, make sure that these updates are synchronized across the system landscape.

#### Background

With the metric provider, you define a metric model using the method `DEFINE_MODEL( )`. This code is executed each time when you activate the metric provider and when the metric provider is imported into target systems. It's important to understand these synchronization events to transport metric providers correctly.

#### Things to Consider

Please keep the following considerations in mind to ensure a consistent metric model for your applications:

- Whenever you add a metric provider to a transport request (because you changed or activated it), also add the implementing class to the transport.

- Whenever you change a metric model in an implementing class, also add the corresponding metric provider to the transport. If you forget to add the metric provider, the metric model synchronization won't be triggered when the transport is imported into another system.
- When you activate a metric provider, the metric model is synchronized. Let's assume you change the behavior of the implementing class: For example, you change the definition of a metric, add a new metric, or you change something else in the metric model of your metric provider. Then you must not only activate the implementing class, but also the metric provider again to trigger a metric model synchronization. After activating the implementing class, log off and on again to the ABAP Development Tools and then activate the metric provider.

## 4.2.9.8.5 Collect Metric Provider Values (Administrator)

As an administrator, you must create an application job to collect metric providers and their measured values for the generic metric store. From the generic metric store, they can be pushed to SAP Cloud ALM or SAP Focused Run.

### Prerequisites

You need an administration role to be able to create application jobs.

You must set up a communication arrangement based on communication scenario SAP\_COM\_0527, so that metrics stored in the generic metric store are automatically pushed to Health Monitoring in SAP Cloud ALM and SAP Focused Run.

### Procedure

1. In the SAP Fiori launchpad, open the *Schedule Metric Provider Collection* app.

2. Choose *Create* to create a new application job.

The job template for collecting metric provider values is already selected.

3. To change job settings, follow the instructions of the in-built documentation of SAP Companion.

#### ⓘ Note

Enter a recurrence that is a multiple of 5 minutes. **Don't** use a recurrence pattern that's shorter than 5 minutes.

4. Schedule the job.

#### ⓘ Note

Make sure that you schedule this job for the development system and all follow-up systems.

## Results

The metric provider values are collected and saved to the generic metric store.

### 4.2.9.8.6 Troubleshooting

If your own metrics for health monitoring aren't collected as expected, check for the possible error sources.

You can check the following:

- Display the log of the application job [Collect Metric Provider Values](#). In the log, search for **\*GSM\*** or the name of the metric provider.  
In the log, you might find error messages saying that metric providers have been banned and the reasons for the bans.  
Background: If the metric provider framework detects any kind of runtime violation such as, for example, too many consecutive dumps, an exceeded runtime limit, or too much memory consumption during the metric value collection, the relevant metric provider is banned for 24 hours. The ban protects the metric measurement of other metric providers. Fix the issues mentioned in the log. After 24 hours, the metric provider framework attempts to execute the provider once more in a tentative mode. If the execution passes without any issues, the ban is lifted, otherwise it's extended by 24 hours.
- If the metric provider is executed, but a metric is missing, you might need to retransport the metric provider to trigger an implicit metric model synchronization.

## Related Information

[Collect Metric Provider Values \(Administrator\) \[page 1197\]](#)

[Considerations for Metric Provider Updates \[page 1196\]](#)

### 4.2.9.9 Accessing ABAP-Managed Data Using SQL Services for Data Integration Scenarios

As a developer in the ABAP environment, you can expose CDS view entities in an ABAP system for external consumption using SQL services.

Using the open database connectivity (ODBC) as a standard API for accessing databases, you can use SQL statements in external analytical tools to access the data of these CDS view entities from your ABAP system and formulate ad-hoc SQL queries on your data. You can also use the exposed CDS view entities from SQL services to set up data replication for further processing of the data and combining it with data from other data sources.

## Why SQL Services?

There are situations where you would like to have external SQL read access to CDS objects owned by the ABAP system. Direct SQL read access to the underlying SAP HANA database of an ABAP system isn't recommended because, for example, ABAP-level security concepts are bypassed and typecasts might not be performed as expected (see also SAP Note [2511210](#)).

Security concerns and other aspects such as authentication and authorization don't arise if you treat the ABAP system itself as a database by accessing the ABAP system directly using the SQL service exposure. In this case, authentication and authorization are done using an ABAP user. Full ABAP SQL semantics apply and even buffering on application server level can be used as well as ABAP-level access control and read access logging.

Compared to the OData interface, the SQL service exposure has the advantage that it allows for unrestricted SQL access to all exposed ABAP CDS view entities. Data from different entities can be joined in an ad-hoc fashion and data can be aggregated for analytical queries.

## Data Integration Patterns and Scenarios

Data access patterns for ABAP-managed data can be categorized into data federation and data replication, as well as privileged access and business user access. Data federation allows for live access to the source system without replicating the data, while data replication involves replicating the data to a second system for local execution. Privileged access enables communication between an SQL-based client and the ABAP system using a communication user. (The communication user is a non-restricted technical power user or represents a technical system.) Business user access applies access controls to limit data access based on authorizations of the business user.

Examples of data consumption scenarios include data consumption using external ODBC clients like Microsoft Excel and LibreOffice, which can work with both privileged and business user access. Another example is data consumption using SAP Datasphere, where data is replicated from an ABAP system to SAP Datasphere for analytical scenarios.

## Related Information

[Data Integration Patterns \[page 1200\]](#)

[Data Consumption Using External ODBC Clients \[page 1201\]](#)

[Data Consumption Using SAP Datasphere \[page 1203\]](#)

## 4.2.9.9.1 Data Integration Patterns

When you set up access to ABAP-managed data using SQL services, you can set up privileged or business user-based data access. As consumption scenarios, there's data federation and replication.

When you set up access to ABAP-managed data to be consumed using SQL services, the following dimensions can be distinguished:

- Data federation vs. data replication
- Privileged access vs. business user access

These dimensions also have interdependencies.

### Data Federation vs. Data Replication

With data federation, data remains in its source system and all requests are forwarded to the source system and executed on the original data. Data federation is used in scenarios where live access to the source system is required, for example, to access the most recent data without a time lag, or to rely on data access control mechanisms inside the source system.

With data replication, data is replicated to a second system and requests are executed locally on the replicated data like, for example, in a data warehouse. Data replication is used where it's more efficient to transfer (business) data from a source to a target system before the data is accessed within that target system.

For more information about the differences between data federation and data replication, see also [Develop Integration Services](#) in the ABAP Cloud documentation.

### Privileged Access vs. Business User Access

#### Privileged Access

With privileged access, the communication between an SQL-based client and the ABAP system is set up for a communication user. Access controls aren't applied on the ABAP side, which means that the communication user can access all data from the CDS view entities for which the communication user is authorized.

Exposure of CDS entities is controlled by service definitions and SQL service bindings. Authorizations are controlled on the level of service bindings or individual CDS view entities. The communication on the ABAP side is configured using communication scenarios and communication arrangements.

In the example used in this documentation, the communication user has read access to all data (order data from all regions).

#### Business User Access

With business user access, communication is established for a business user. Access controls for the exposed CDS view entities are active and the business user can only access data from the exposed CDS view entities filtered by the business user's authorizations.

Exposure of CDS entities is controlled by service definitions and SQL service bindings. Authorizations are controlled on the level of service bindings and the authorizations that are referenced in access controls. Authorizations are assigned using IAM apps, restriction types, and business catalogs.

In the example in this documentation, a restriction type is used to limit the access of the business user to order data from one region only.

## Dependencies Between Dimensions

The following table shows how these dimensions can be combined:

	Data Federation	Data Replication
Privileged Access	Supported	Supported
Business User Access	Supported	Not supported

## Examples

In the documentation here, you can find examples of different data access patterns: For example, the data consumption using the external ODBC clients Microsoft Excel and LibreOffice are data federation scenarios that work with privileged and business user access. Data replication is currently only supported using SAP Datasphere as replication middleware.

## Related Information

[Data Consumption Using External ODBC Clients \[page 1201\]](#)

[Data Consumption Using SAP Datasphere \[page 1203\]](#)

## 4.2.9.9.2 Data Consumption Using External ODBC Clients

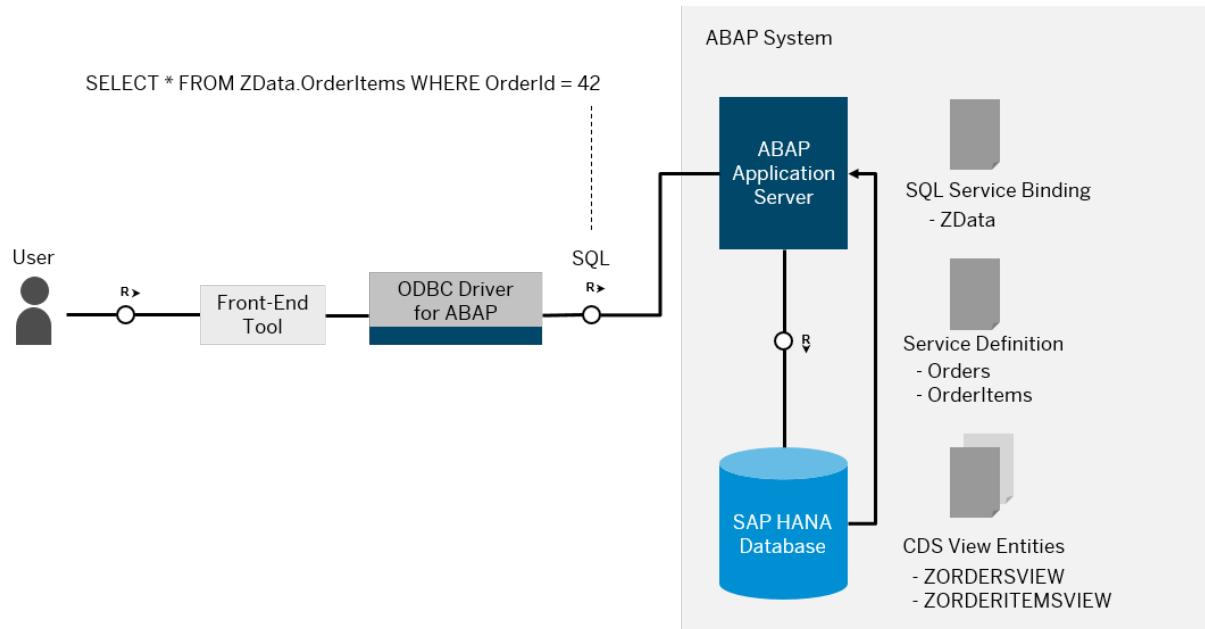
As a developer in the ABAP environment, you can access CDS view entities in an ABAP system using SQL and the open database connectivity (ODBC), a standard API for accessing databases. As a result, you can use SQL

statements in external analytical tools to access data in database tables (via CDS view entities) that reside in the ABAP environment.

## Architecture

To access CDS view entities from external ODBC-based tools, you use the SQL service binding and the ODBC driver for ABAP as follows:

- The ODBC driver for ABAP implements the ODBC API, which can be used by ODBC-enabled front-end tools to execute SQL queries on the SAP Application Server ABAP. Such a front-end tool could be a standard tool capable of loading an ODBC driver, for example, Microsoft Excel. The ODBC driver for ABAP can also be used in multiple programming languages. ODBC is primarily a C API, but other languages like node.js, Python, or Perl also provide plugins for ODBC drivers.
- Instead of directly querying the underlying ABAP-owned database objects using the SQL layer of the ABAP system database, all SQL requests are routed through the ABAP Application Server. They behave just like they would if executed from within an ABAP program.
- You create CDS view entities on top of the tables that you want to expose.
- The SQL service binding type serves to expose CDS view entities as read-only to ODBC-based SQL clients. With the SQL service, you can use an access mechanism of the ABAP Application Server that provides SQL-level access to published ABAP-managed database API objects like tables and CDS views via CDS view entities for system-external consumers.



The ODBC driver for ABAP supports the use of a communication user in the ABAP system with privileged access (no access controls applied). Only read access to the exposed ABAP CDS objects is allowed.

In addition to the communication user, the ODBC driver for ABAP also supports the use of a business user with a browser-based logon. The ABAP system forwards the user's logon request to the configured identity provider to handle it. Authorizations for business users to view data can be limited using restriction types.

The ODBC driver for ABAP is available on Windows and Linux.

You can access the data as read-only using an external ODBC-based client tool, including loading the data into the client tool and searching for data using embedded SQL queries. The ODBC-based client can be any client tool capable of loading an ODBC driver, such as Microsoft Excel or LibreOffice.

## Additional Information

This documentation leads you through the necessary development, installation, and configuration for the consumption of the SQL service and ODBC driver. If you prefer to follow a tutorial instead, you can also use the following (for a communication user with privileged access only):

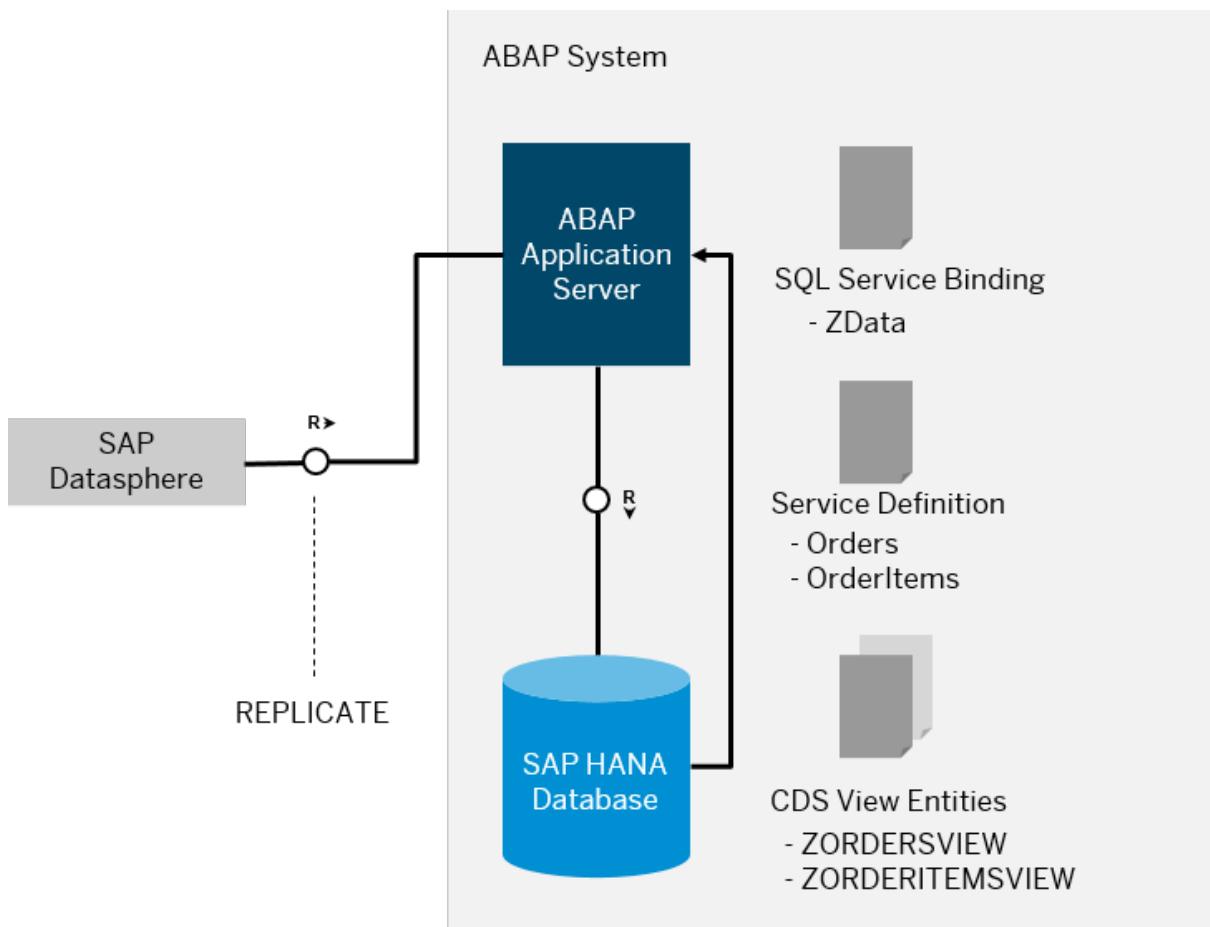
<https://developers.sap.com/tutorials/abap-environment-abap-sql.html>

### 4.2.9.9.3 Data Consumption Using SAP Datasphere

You can consume ABAP-managed data using SAP clients such as SAP Datasphere.

#### Architecture

In this scenario, you use SAP Datasphere to retrieve data from your ABAP system. From a technical perspective, SAP Datasphere plays the role of a client and triggers replication of data from the ABAP system to your SAP Datasphere instance. The replicated data then forms the basis for analytical scenarios inside SAP Datasphere.



To access data from your ABAP system in SAP Datasphere, you use SQL service bindings in the ABAP systems for the data exposure and the replication flows provided by SAP Datasphere:

- You create CDS view entities on top of the tables or CDS views that you want to expose.
- The SQL service binding type serves to expose CDS view entities. With the SQL service, you can use an access mechanism of the ABAP application server that provides SQL-level access to published ABAP-managed database API objects like tables and CDS views via CDS view entities for replication.
- The replication management service of SAP Datasphere is used to replicate selected CDS view entities to SAP Datasphere.

In the replication scenario, the use of a communication user in the ABAP system with privileged access is supported (no access controls applied). Only read access to the exposed CDS view entities is allowed.

#### 4.2.9.9.4 Prerequisites and Constraints

Check the prerequisites and constraints before you start using the SQL service for accessing ABAP-managed data.

##### [Prerequisites \[page 1205\]](#)

Check whether you meet all prerequisites for using the SQL service.

##### [Constraints \[page 1206\]](#)

There are some constraints regarding functionality that can be exposed using an SQL service binding.

#### [Additional Prerequisites and Constraints for Replication \[page 1207\]](#)

If you want to use an SQL service for replication, additional prerequisites and constraints apply.

### 4.2.9.4.1 Prerequisites

Check whether you meet all prerequisites for using the SQL service.

Make sure that the following general requirements are met:

- You have a developer user in the ABAP environment.
- You have ABAP Development Tools (ADT) installed.
- You have an ABAP Cloud project configured in ADT and connected to the ABAP system.

To be able to run the ODBC driver for ABAP, the following prerequisites must be met:

- Make sure that you use a 64-bit application as an ODBC client because the ODBC driver for ABAP is only available as a 64-bit driver.
- **Linux only:** A 64-bit Linux version is required because the ODBC driver for ABAP is a 64-bit ODBC driver. To define ODBC data source names (DSN) for the ODBC driver for ABAP on Linux, you must install the unixODBC software package on your Linux system. This software package also provides some simple command-line tools to test an ODBC connection and to execute SQL queries.
- **Linux only:** Set the environment variable where the driver files reside, such as:  
`export LD_LIBRARY_PATH=/home/<user_name>/odbc-abap`  
To avoid that you need to extend `LD_LIBRARY_PATH` manually every time the ODBC driver is used, consider adding this setting to the Linux logon scripts for the users using the ODBC driver from this location.
- **Microsoft Windows only:** The ODBC driver requires a C/C++-runtime (VCredist). Make sure that the latest version of the VCredist package is installed. For more information about VCredist, see SAP Note [1553465](#).
- If you want to use the open-source software package LibreOffice as an external ODBC client, make sure that you have LibreOffice installed.
- **Business user with browser logon only:** If you want to use the access scenario for business users with browser logon, you need at least ODBC driver version 01, patch level 2.

### Related Information

#### [Data Integration Patterns \[page 1200\]](#)

## 4.2.9.4.2 Constraints

There are some constraints regarding functionality that can be exposed using an SQL service binding.

Only CDS view entities can be exposed, for the following reasons:

- Only one name (CDS name equals SQL name)
- No table functions for views with parameters (better performance)
- Stricter checks regarding, for example, completeness of currency/unit references
- Harmonized client handling

These CDS view entities must meet **all** of the following criteria:

- They are not part of a transactional business object (BO).
  - Reason: Access on SQL-/DB-level doesn't consider the transactional buffer, so not all relevant data is considered.
  - Criteria:
    - Not part of a ROOT-COMPOSITION relationship OR
    - They don't contain the header annotation `@ObjectModel.transactionalProcessingEnabled`
- They don't represent an analytical query.
  - Reason: Access on SQL-/DB-level doesn't compute the correct results as query contains features that can only be interpreted by an OLAP engine.
  - Criteria: They don't contain the header annotation `@Analytical.query`.
- They don't represent an (enterprise) search model.
  - Reason: Access on SQL-/DB-level doesn't compute the correct results as only the enterprise search engine can correctly interpret/consume the view model.
  - Criteria: They don't contain the header annotation `@EnterpriseSearch.enabled`.
- They aren't implemented via a custom query.
  - Reason: Access on SQL-/DB-level doesn't compute the correct results as the query is implemented on ABAP-level.
  - Criteria: They don't contain the header annotation `@ObjectModel.query.implementedBy`.
- They aren't implemented via an ABAP-implemented analytical cube.
  - Criteria: They don't contain the header annotation `@Analytics.readClassName`.
- They are no private views.
  - Reason: Even if access on SQL-/DB-level computed the correct results, these views should only be called locally in the system.
  - Criteria: They don't contain the header annotation `@VDM.private`.
- They are no extension include views.
  - Reason: Even if access on SQL-/DB-level computed the correct results, these views should only be called locally in the system.
  - Criteria: They don't contain the header annotation `@VDM.viewType: #EXTENSION`.
- They don't represent derivation functions.
  - Reason: Even if access on SQL-/DB-level computed the correct results, these views should only be called locally in the system.
  - Criteria:

- No header annotation `@VDM.viewType: #DERIVATION_FUNCTION`
- No header annotation `@ObjectModel.derivationFunction`
- They don't contain elements that meet one of the following criteria:
  - Virtual elements
  - Criteria:
    - They are not defined via the keyword `VIRTUAL` or:
    - They don't contain the header annotations `@ObjectModel.virtualElement` or `@ObjectModel.virtualElementCalculatedBy`.
  - Private elements
    - Criteria: They don't contain the header annotation `@Consumption.hidden`.
  - Derived elements
    - Criteria: They don't contain the header annotation `@ObjectModel.value.derivedFrom`.

#### 4.2.9.9.4.3 Additional Prerequisites and Constraints for Replication

If you want to use an SQL service for replication, additional prerequisites and constraints apply.

The following generally applies for replication using an SQL service:

- CDS view entities with parameters can never be replicated.
- An initial load (complete replication of all records) is possible without further restrictions.
- Every CDS view entity that is used for **delta** replication must declare this intention in its header via the annotation `@DataIntegration.deltaReplication.intended: true`.

You must perform an initial load again if **all** of the following applies:

- A table structure has been changed, which has also affected the primary key.
- The changed primary key is used in a CDS view entity that is part of a replication scenario.

In this case, a delta replication is not sufficient, but you must re-initialize the process with a full replication of the records.

For delta replication on CDS view entities, additional constraints apply. For more information and a full list of constraints, see the Knowledge Transfer Documentation for the annotation `@DataIntegration.deltaReplication.intended: true`, which can be called up in ABAP Development Tools.

## **4.2.9.9.5 Developing and Exposing an SQL Service in the ABAP System**

Before you can access database tables from external ODBC-based tools, you must start with some preparatory steps such as developing and exposing an SQL service in the ABAP system.

### **Context**

When you want to enable the access to ABAP-managed data from external ODBC clients, you must start with some preparatory steps. These steps include the creation of CDS view entities for the tables for which you want to access, a service definition, and an SQL-typed service binding. Optionally, you can also create your own test tables. Test tables are used in this documentation as examples.

Depending on your access scenario (privileged data access with a communication user or data access with a business user), you must perform different additional steps.

### **Privileged Access with a Communication User**

For accessing database tables using a communication user, you must also create a communication scenario and a communication system in ABAP Development Tools. An administrator creates a corresponding communication arrangement and user. The assumption in this documentation is that the communication user gets privileged data access.

### **Access with a Business User**

For accessing database tables using a business user, this documentation also shows how to configure data access where access is limited to orders for a specific region. For this scenario, you must create an authorization object, access control, an IAM app, and a business catalog in ABAP Development Tools. An administrator creates a business role and assigns a business user.

### **Related Information**

[Data Integration Patterns \[page 1200\]](#)

## 4.2.9.9.5.1 Creating a Data Model and an SQL Service

Before you set up the data access for ODBC-based clients, you must create a data model and an SQL service in ABAP Development Tools.

In this documentation, you can find instructions how to create a few tables with sample data. These tables are used as examples to show how data access for ODBC-based clients works.

In addition, you need CDS view entities for your tables, a service definition, and an SQL-typed service binding.

### 4.2.9.9.5.1.1 Creating a Data Element

As part of the development artefacts for testing, create a data element `ZREGION`.

#### Context

This data element `ZREGION` is used as an example to illustrate data access for business users based on regions. (The data element is also part of the example data for privileged access, but won't be used.)

As part of the development artefacts, you also create a table `ZORDERS` (see [Creating and Filling Test Tables \[page 1210\]](#)). To use instance-level restrictions on the data, you extend the `ZORDERS` table by adding a region field. You can then later restrict access based on the region of an order.

#### Procedure

- Create a data element `ZREGION`, based on a domain `ZREGIONDOM` with a value table `ZREGIONS`.

For example, your code can look like the following:

##### ↔ Sample Code

```
@EndUserText.label : 'REGIONS'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #A  
@AbapCatalog.dataMaintenance : #RESTRICTED  
define table zregions {  
    key region : zregion not null;  
}
```

## 4.2.9.9.5.1.2 Creating and Filling Test Tables

As an example, let's create and fill some test tables to illustrate how database tables can be accessed from external ODBC-based tools.

As an example, the demo table entities ZORDERS and ZORDERITEMS are used in this documentation.

The definition of the ZORDERS table in ABAP Development Tools (ADT) can look as follows:

### ↳ Sample Code

```
@EndUserText.label : 'ORDERS'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #A  
@AbapCatalog.dataMaintenance : #RESTRICTED  
define table zorders {  
    key id      : abap.numc(10) not null;  
    creationdate : abap.datn;  
    region      : zregion;  
}
```

Here's the table for order items:

### ↳ Sample Code

```
@EndUserText.label : 'ORDER ITEMS'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #A  
@AbapCatalog.dataMaintenance : #RESTRICTED  
define table zorderitems {  
    key orderid : abap.numc(10) not null;  
    key pos      : abap.int4 not null;  
    item       : abap.char(100) not null;  
    amount     : abap.int4 not null;  
}
```

You can also create some test data in the tables using the following ABAP sample code:

### ↳ Sample Code

```
class zcl_fill_orders definition  
public  
final  
create public.  
public section.  
    interfaces if_oo_adt_classrun.  
protected section.  
private section.  
endclass.  
class zcl_fill_orders implementation.  
method if_oo_adt_classrun~main.  
    data: lt_regions type table of zregions.  
    delete from zregions.  
    lt_regions = value #(  
        ( region = 'A' )  
        ( region = 'B' )  
    ).  
    insert zregions from table @lt_regions.  
    out->write( sy-dbcnt ).
```

```

data: lt_orders type table of zorders.
delete from zorders.
lt_orders = value #(
    ( id = '1' creationdate = '20210801' region = 'A' )
    ( id = '2' creationdate = '20210802' region = 'A' )
    ( id = '3' creationdate = '20210803' region = 'B' )
    ( id = '4' creationdate = '20210804' region = 'B' )
).
insert zorders from table @lt_orders.
out->write( sy-dbcnt ).
data: lt_orderitems type table of zorderitems.
delete from zorderitems.
lt_orderitems = value #(
    ( orderid = '1' pos = '1' item = 'Apple' amount = '5' )
    ( orderid = '1' pos = '2' item = 'Banana' amount = '5' )
    ( orderid = '1' pos = '3' item = 'Orange Juice' amount = '2' )
    ( orderid = '2' pos = '1' item = 'Orange' amount = '10' )
    ( orderid = '2' pos = '2' item = 'Apple' amount = '5' )
    ( orderid = '3' pos = '1' item = 'Bottle Water' amount = '5' )
    ( orderid = '4' pos = '1' item = 'Bottle Wine' amount = '2' )
    ( orderid = '4' pos = '2' item = 'Apple' amount = '5' )
    ( orderid = '4' pos = '3' item = 'Orange' amount = '3' )
).
insert zorderitems from table @lt_orderitems.
out->write( sy-dbcnt ).
endmethod.
endclass.

```

To be able to use regions to control data access, the ABAP class for filling the tables also fills a ZREGIONS table and includes a region in the orders.

### 4.2.9.9.5.1.3 Creating CDS View Entities for Tables

The example code gives you an idea how you create CDS view entities for data integration using an SQL service.

#### Context

In this example, you learn how you can create CDS view entities on top of the tables that you want to expose, using the ADT wizard. Only CDS view entities can be exposed to consumers using the SQL service.

#### Procedure

1. In the project explorer in ABAP Development Tools, right-click on the table and choose *New Data Definition* from the context menu.
2. Fill out the data requested in the following popup and choose *Next*.
3. On the next screen, choose *Define View Entity* and *Finish*.
4. Activate the new CDS view entities.

## Results

The definitions can look as follows:

### ↳ Sample Code

```
@AbapCatalog.viewEnhancementCategory: [ #NONE ]
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Orders'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
@DataIntegration.deltaReplication.intended: true
define view entity ZORDERSVIEW as select from zorders {
    key id as Id,
    creationdate as CreationDate,
    region as Region
}
```

### ↳ Sample Code

```
@AbapCatalog.viewEnhancementCategory: [ #NONE ]
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'ORDER ITEMS'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
@DataIntegration.deltaReplication.intended: true
define view entity ZORDERITEMSVIEW as select from zorderitems
    association [1..1] to zorders as _order on $projection.Orderid = _order.id
    {
        key orderid as Orderid,
        key pos as Pos,
        item as Item,
        amount as Amount,
        _order
    }
```

### ⓘ Note

The original table columns have been renamed to use mixed-case names.

For business user access using a region field, the ZORDERSVIEW CDS view entity also includes the region field of the ZORDERS table. For privileged access, the region field is not needed.

For business user access, in the ZORDERITEMSVIEW CDS view entity, you make the region field of the orders accessible using an association \_order.

For privileged access, no access control is needed, here you can replace

```
@AccessControl.authorizationCheck: #CHECK by @AccessControl.authorizationCheck:
#NOT_REQUIRED.
```

For business user access, there's a warning that no access control exists. You create this access control when you expose the SQL service for business user access.

The annotation `@DataIntegration.deltaReplication.intended: true` is only required for data replication scenarios (delta replication).

#### 4.2.9.9.5.1.4 Creating a Service Definition and an SQL-Typed Service Binding

To expose CDS view entities using the SQL service, you must create a service definition and a corresponding service binding of type SQL1.

##### Procedure

1. To define a new service definition, right-click on the CDS views that you created in the project explorer and choose [New Service Definition](#) in the context menu.
2. Enter a name for the new service definition.
3. Open the new service definition, add the second view, and add alias names for the CDS view entities.

Adding an alias is not required but recommended because with an alias, you can exchange the CDS view entity behind the alias but keep the contract to the consuming application stable.

The service definition can look as follows, for example:

##### ↔ Sample Code

```
@EndUserText.label: 'SERVICE DEF'
define service Z_SERVICE_DEF_SQL provider contracts sql {
    expose ZORDERSVIEW as Orders;
    expose ZORDERITEMSVIEW as OrderItems;
}
```

Again, mixed-case names have been chosen for the alias names.

4. Activate the service definition.
5. To create a service binding, right-click on the service definition and choose [New Service Binding](#).
6. In the following popup, enter a name and description, for example, `ZDATA`.

You can use a mixed-case name for the service binding. The service binding name acts as the schema name for external consumers.

7. As binding type, select [SQL – Web API](#) and choose [Next](#) and [Finish](#).
8. In the [Enabled Operations](#) area, select all access types that you want to allow on the service.

`SELECT` is selected by default and enables federated access. `REPLICATE` enables replicated access.

9. Activate the service binding.

## 4.2.9.9.5.2 Exposing the SQL Service for Data Federation and Replication with Privileged Access

To enable privileged data access for communication users using the SQL service exposure, you must control access to data using communication scenarios, communication systems, and communication arrangements.

### 4.2.9.9.5.2.1 Creating a Communication Scenario with Object Privileges

To be able to use a communication user in the ABAP system to access the service binding, you must create a communication scenario in ABAP Development Tools.

#### Context

This communication scenario is the basis of a communication arrangement in the ABAP environment that you also need.

#### Procedure

1. In the project explorer of the ABAP Development Tools, right-click on your project and choose  [New](#)  [Other ABAP Repository Object](#).
2. In the popup, enter **Communication** in the search field and then choose **Communication Scenario** from the results.
3. Enter a name and description for the new communication scenario and choose **Next** and **Finish**.
4. In the communication scenario definition, choose the **Inbound** tab.
5. To use user/password authentication, choose **Basic** as supported authentication method. Alternatively, you can use authentication with X.509 client certificates by choosing **X.509**.
6. In the **Inbound Services** screen area, choose **Add...**, enter the **S\_PRIVILEGED\_SQL1** service, and choose **Finish**.

The **S\_PRIVILEGED\_SQL1** inbound service is a preconfigured service for the privileged access to CDS view entities, that is, no DCLs are applied. (DCL stands for data control language. It provides an access control mechanism to restrict the results returned by the CDS view from the database according to conditions.)

7. To add authorizations and enable access to the service binding, choose the **Authorizations** tab.
8. Under **Authorization Objects**, choose **Insert** and add the **s\_SQL\_VIEW** authorization object.
9. Choose the newly added authorization object and fill out the authorizations in the details, such as the following:

Field	Value
SQL_SCHEMA	The name of the service binding (exposed as a schema) that you want to grant access to
SQL_VIEW	Comma-separated list of all views to which you want to grant access or * (if you want to allow access to all views in the service definition)
SQL_VIEWOP	Choose at least one option: <i>SELECT</i> or <i>REPLICATE</i> . SELECT grants read access to and REPLICATE allows replication on the specified views.

**① Note**

You can allow replication on a view here, but the replication might still not be technically possible.

We recommend that you create separate communication scenarios for SELECT and REPLICATE to give administrators more fine-grained settings.

10. Save your entries and choose *Publish Locally* to publish the communication scenario in the development system.

After publication, you can create a communication arrangement based on the communication scenario.

## Related Information

[Communication Management \[page 883\]](#)

### 4.2.9.9.5.2.2 Creating a Communication System and a Communication User

An administrator must create a communication system and a communication user for accessing ABAP-managed data from external ODBC-based clients.

## Procedure

1. Log on to the SAP Fiori launchpad of the ABAP environment as an administrator.
2. Under *Communication Management*, choose the *Communication Systems* app.
3. Choose *New*, enter a system ID and name for the communication system, and choose *Create*.
4. In the newly created communication system, select the checkbox *Inbound Only*.
5. Under *Users for Inbound Communication*, choose the + button and then *New User*.

6. Enter a user name (for example, **SQL\_CLIENT\_USER**), a description, and a password.  
Remember the user name and password for later.
7. Choose *Create*.
8. To add the new user to the communication system, choose *OK* in the dialog.  
The user is now listed in the screen area *Users for Inbound Communication*.
9. Choose *Save* to finish the creation of the communication system.

### 4.2.9.9.5.2.3 Creating a Communication Arrangement for Exposing the SQL Service

As a last step of the back-end configuration for exposing an SQL service in the ABAP system, an administrator must create a communication arrangement.

#### Context

The arrangement links the communication scenario that was created in ABAP Development Tools with the communication system and communication user.

#### Procedure

1. Log on to the SAP Fiori launchpad of the ABAP environment as an administrator.
2. In the SAP Fiori launchpad, under *Communication Management*, choose *Communication Arrangements*.
3. Choose *New*.
4. In the following popup, enter the communication scenario that you created before and choose *Create*.
5. In the newly created communication arrangement, enter the communication system that you created before (see [Creating a Communication System and a Communication User \[page 1215\]](#)).  
This step completes the link between communication scenario and communication system. The system automatically adds the communication user.
6. Take note of the service URL.  
It's something like `https://<hostname>/sap/bc/sql/sql1/sap/S_PRIVILEGED`.
7. Choose *Save*.

#### Results

You've now finished all preparation tasks in the ABAP system and can access the exposed objects as an SQL service.

## **4.2.9.9.5.3 Exposing the SQL Service for Data Federation with Business User Access**

To enable data access for business users using ODBC-based clients, you must control access to data using IAM apps, restriction types, and business catalogs.

With the example used in this documentation, you will learn how to restrict access to order data based on an order's region.

### **4.2.9.9.5.3.1 Creating an Authorization Field and Object**

Create an authorization field and object to provide access control based on a field in the relevant CDS views.

#### **Context**

There's a region field available in the example CDS views. Based on this example, let's create an authorization field and object.

For more information about creating authorization objects and fields, see the documentation for ABAP Development Tools.

#### **Procedure**

1. In ABAP Development Tools, create an authorization field.

In the example used in this documentation, create an authorization field `ZREGION` for the region, based on the `ZREGIONS` table.

2. Create an authorization object with an activity field and the authorization field that you just created.

In the example used here, this is an authorization object `ZREGIONMGT` with the authorization field `ZREGION`.

## 4.2.9.9.5.3.2 Creating Access Control

To make sure that no unauthorized users access data, create access control for your CDS views.

### Context

In the example used in this documentation, you create two access controls – one for each view – and use the aspect `pfcg_auth` clause to access the region field in the authorization object. For the sake of simplicity, we don't check the activity, that is, every activity is valid to view the records of a region.

### Procedure

Create the following access controls:

#### Sample Code

```
@EndUserText.label: 'ORDERS DCL'
@MappingRole: true
define role ZORDERSVIEW_DCL {
    grant
        select
        on
            ZORDERSVIEW
            where
                (region) = aspect pfcg_auth(ZREGIONMGT, ZREGION);
}
```

#### Sample Code

```
@EndUserText.label: 'ORDER ITEMS DCL'
@MappingRole: true
define role ZORDERITEMSVIEW_DCL {
    grant
        select
        on
            ZORDERITEMSVIEW
            where
                (_order.region) = aspect pfcg_auth(ZREGIONMGT, ZREGION);
}
```

In the second access control, you use the `_order` association to get access to the order's region field.

## 4.2.9.9.5.3.3 Creating an IAM App and a Restriction Type

To provide a data access for a business user in an ODBC scenario, you must create a restriction type and an IAM app in ABAP Development Tools (ADT).

### Prerequisites

The data that the business user needs to access already exists and there's a corresponding authorization object and authorization field available.

### Context

As a developer, you create an IAM app and a restriction type based on the available authorization field to provide access control for the data. For more information about how to create restriction types and IAM apps, see the ADT documentation.

### Procedure

1. In ABAP Development Tools, create an IAM app called, for example, `ZDATA_IAM`.
2. On the *Services* tab for the IAM app, add the SQL service that you created.  
In the example used for this documentation, this is the `ZDATA` SQL service.
3. On the *Authorizations* tab, add the authorization object that you want to use.  
In the example used for this documentation, this is the `ZREGIONMGT` authorization object.
4. For the `ACTVT` authorization field, choose the checkboxes *Add or Create*, *Change*, *Display*, and *Delete*.  
For all other authorization fields (for example, `ZREGION` in this documentation), leave the settings unchanged. You will model access control using a restriction type.
5. To create a restriction type, open the authorization object that you want to use (for example, `ZREGIONMGT` in this documentation).
6. In the *What's next* section, choose *Create restriction type based on the authorization object*.  
In the example used here, let's call the new restriction type `ZREGIONMGT_RSTR`.

#### 4.2.9.9.5.3.4 Creating a Business Catalog

To provide data access to business users using ODBC-based clients, you create a business catalog and assign an IAM app to it.

#### Context

For more information about creating business catalogs, assigning IAM apps and restriction types, see the ABAP Development Tools documentation.

#### Procedure

1. In ABAP Development Tools, create a business catalog, for example, called `ZDATA_BC`.
2. On the [Apps](#) tab, assign the IAM app that you have created before (see [Creating an IAM App and a Restriction Type \[page 1219\]](#)).
3. On the [Restriction Types](#) app, assign the restriction type that you have created before.
4. Choose the *Read* checkbox.
5. Choose [Publish Locally](#) to make the business catalog visible in the development system for testing.

To make the business catalog and all other development objects that you created available for productive use, transport them to the relevant production system.

#### 4.2.9.9.5.3.5 Creating a Business Role and Assigning it to a Business User

To enable business users to access ABAP-managed data from ODBC clients, an administrator must create an appropriate business role and assign it to the business user.

#### Prerequisites

##### ① Note

As opposed to the previous steps in this documentation that were performed by a developer, you now need an administrator role.

An appropriate business catalog for the ODBC client scenario has been created by a developer (for example, `ZDATA_BC`, see [Creating a Business Catalog \[page 1220\]](#)).

## Procedure

1. Create a new business role `BR_ZDATA` and assign the business catalog `ZDATA_BC` to it.
2. On the *Assigned Business Users* tab, choose *Maintain Restrictions*.

In the example used in this documentation, the restriction type *Region Management* is shown.

3. Add a value, for example, add the value `A` for the `REGION` field.
4. Go back to the role, add a business user to it, and save the role.

## Results

The assigned business user will now be able to see data for region A only.

### 4.2.9.9.6 Installation and Configuration of the ODBC Driver for ABAP (Data Federation Only)

To be able to use an ODBC-based client to access data in the exposed CDS view entities of the ABAP environment, you must install the ODBC driver and create an ODBC data source.

#### ⓘ Note

If you use the SQL service in a **replication** scenario with SAP Datasphere only, the installation of the ODBC driver for ABAP is not needed.

Proceed as follows:

1. Install the ODBC driver on your operating system (see [Installing the ODBC Driver \[page 1221\]](#)).
2. Create an ODBC data source on your operating system (see [Creating an ODBC Data Source on Windows \(Privileged Access\) \[page 1222\]](#) or [Creating an ODBC Data Source on Linux \(Privileged Access\) \[page 1223\]](#)).

#### 4.2.9.9.6.1 Installing the ODBC Driver

Get the ODBC driver from the SAP Support Portal and install it.

## Context

You need the ODBC driver for ABAP to be able to access exposed CDS entities from an ODBC-enabled client.

## Procedure

1. Go to the SAP Support Portal at <https://support.sap.com/>.
2. Choose *Software Downloads*.  
You might need to log on now.
3. Search for the components *ODBC DRIVER FOR ABAP 1.0* and *SAPCRYPTOLIB*.
4. Download the correct SAR files for your operating system. In addition, you can download SAPCAR to be able to unpack the SAP files.
5. Choose a directory as your ODBC driver location and unpack the SAR files there.
6. **Windows only:** To install the ODBC driver for ABAP, start the installer.  
For Linux, unpacking the SAR files is sufficient.

## Results

The ODBC driver is now ready to be used in ODBC data sources.

### 4.2.9.9.6.2 Creating an ODBC Data Source on Windows (Privileged Access)

The *ODBC Data Sources (64 bit)* tool is part of a standard Windows installation. In this tool, you can create ODBC data sources and assign a data source name (DSN) to a newly created data source.

## Procedure

1. Launch the Windows ODBC data source administrator tool.
2. Choose the *User DSN* or *System DSN* tab, choose *Add*, choose the ODBC driver for ABAP as ODBC driver, and choose *Finish*.  
The DSN setup dialog of the ODBC driver for ABAP starts.
3. In this dialog, choose a DSN and fill in the driver-specific parameters that are described in more detail in SAP Note [3076454](#).

#### ⓘ Note

You can derive the host name and the service path from the service URL that you wrote down when you created the communication arrangement.

The user name (for example, `SQL_CLIENT_USER`) that you created in the communication system is automatically the alias name for the generated ABAP user name. Therefore, switch the user type to alias.

## 4.2.9.9.6.3 Creating an ODBC Data Source on Linux (Privileged Access)

With the settings described here, you create an ODBC data source and DSN for a communication user.

### Context

The unixODBC software package provides an ODBC driver manager that can read from multiple configuration files. You can create configuration files that define the location of the ODBC driver and DSNs.

### Procedure

1. List the locations of the configuration files by executing the command `odbcinst -j`.

You get a result such as the following, for example:

#### Sample Code

```
> odbcinst -j
unixODBC 2.3.6
DRIVERS.....: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc. [MYDSN]>
FILE DATA SOURCES..: /etc/unixODBC/ODBCDataSources
USER DATA SOURCES..: /home/myuser/.odbc.ini
SQLULEN Size.....: 8
SQLLEN Size.....: 8
SQLSETPOSIROW Size.: 8
```

2. To define both the driver and a user-specific DSN, create the file `/home/myuser/.odbc.ini`.

You can create a system-wide driver definition and DSN in a similar way.

3. In the `/home/myuser/.odbc.ini` file, insert the DSN-specific connection parameters for the ODBC driver for ABAP as described in SAP Note [3076454](#).

For example, these parameters can look as follows:

#### Sample Code

```
[MYDSN]
; this is a comment
Driver=/home/<myuser>/ODBC_driver_for_ABAP/ODBC_driver_for_ABAP.so
HOST=<hostname>
PORT=443
CLIENT=100
LANGUAGE=EN
SERVICEPATH=/sap/bc/sql/sql1/sap/S_PRIVILEGED
TrustAll=true
CryptoLibrary=/home/<myuser>/ODBC_driver_for_ABAP/libsapcrypto.so
UidType=alias
```

In this example, the driver definition was included in the `.odbc.ini` file using the `driver` keyword. As connect user for the test system, you can use the user `SQL_CLIENT_USER` that you created as an alias user in the ABAP environment. It's also possible to store the `UID/PWD` properties in the `.odbc.ini` file. However, we don't recommend the use of a file for security reasons. In the example, for the sake of simplicity, no PSE file was created and the `TrustAll=True` property was used instead.

4. Use the unixODBC tools `isql` or `isql` to test the ODBC connection.

If everything is configured correctly, you can see the following in the `isql` tool, for example:

#### Sample Code

```
> iusql MYDSN SQL_CLIENT_USER <password> -v
+-----+
| Connected!
|
| sql-statement
| help [tablename]
| quit
|
+-----+
SQL>
```

## 4.2.9.9.6.4 Creating an ODBC Data Source on Windows (Business User Access)

You create an ODBC data source and DSN for a business user. To enable a business user to log on to the ABAP system with user name and password, you configure the browser-based logon method in the ODBC DSN.

### Context

The [ODBC Data Sources \(64 bit\)](#) tool is part of a standard Windows installation. In this tool, you can create ODBC data sources and assign a data source name (DSN) to a newly created data source.

In the example used in this documentation, you need the ODBC data source for a business user that is provided data access to the ABAP system.

### Procedure

1. Launch the Windows ODBC data source administrator tool.
2. Choose the [User DSN](#) or [System DSN](#) tab, choose [Add](#), choose the ODBC driver for ABAP as ODBC driver, and choose [Finish](#).

The DSN setup dialog of the ODBC driver for ABAP starts.

3. In this dialog, choose a DSN and fill in the driver-specific parameters that are described in more detail in SAP Note [3076454](#).

### Note

ServicePath must point to the SQL service that you created (in the example used here, ZData):  
ServicePath=/sap/bc/sql/sql1/sap/zdata.

Don't forget to configure the browser-logon method using AuthenticationType and AuthenticationURL:

AuthenticationType=Browser

AuthenticationURL=https://<ui host name>/sap/bc/sec/reentrance

<ui host name> is the standard host name also used in the SAP Fiori Launchpad.

## 4.2.9.9.6.5 Creating an ODBC Data Source on Linux (Business User Access)

With the settings described here, you create an ODBC data source and DSN for a business user. To enable a business user to log on to the ABAP system with user name and password, you configure the browser-based logon method in the ODBC DSN.

### Context

The unixODBC software package provides an ODBC driver manager that can read from multiple configuration files. You can create configuration files that define the location of the ODBC driver and DSNs.

In the example used in this documentation, you need the ODBC data source for a business user that is provided data access to the ABAP system.

### Procedure

1. List the locations of the configuration files by executing the command odbcinst -j.

You get a result such as the following, for example:

### Sample Code

```
> odbcinst -j
unixODBC 2.3.6
DRIVERS.....: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc.. [MYDSN]>
FILE DATA SOURCES...: /etc/unixODBC/ODBCDataSources
USER DATA SOURCES...: /home/myuser/.odbc.ini
SQLULEN Size.....: 8
SQLLEN Size.....: 8
SQLSETPOSIROW Size.: 8
```

2. To define both the driver and a user-specific DSN, create the file `/home/myuser/.odbc.ini`.

You can create a system-wide driver definition and DSN in a similar way.

3. In the `/home/myuser/.odbc.ini` file, insert the DSN-specific connection parameters for the ODBC driver for ABAP as described in SAP Note [3076454](#). In addition, you configure the browser-logon method using `AuthenticationType` and `AuthenticationURL`.

For example, these parameters can look as follows:

#### ↔ Sample Code

```
[MYDSN]
; this is a comment
Driver=/home/<myuser>/ODBC_driver_for_ABAP/ODBC_driver_for_ABAP.so
HOST=<hostname>
PORT=443
CLIENT=100
LANGUAGE=EN
SERVICEPATH=/sap/bc/sql/sql1/sap/zdata
TrustAll=true
CryptoLibrary=/home/<myuser>/ODBC_driver_for_ABAP/libsapcrypto.so
AuthenticationType=Browser
AuthenticationURL=https://<ui host name>/sap/bc/sec/reentrance
UID=dummy
PWD=dummy
```

In this example, the driver definition was included in the `.odbc.ini` file using the `driver` keyword. In the example, for the sake of simplicity, no PSE file was created and the `TrustAll=True` property was used instead. The entries `UID=dummy` and `PWD=dummy` were added so that you are not forced to enter a user and password in the browser that are required by the unixODBC tool `issql` for testing, but are not needed by the ODBC driver.

`<ui host name>` is the standard host name also used in the SAP Fiori Launchpad.

4. Use the unixODBC tools `isql` or `issql` to test the ODBC connection.

If everything is configured correctly, you can see the following in the `issql` tool, for example:

#### ↔ Sample Code

```
> issql MYDSN -v
+-----+
| Connected!
|
| sql-statement
| help [tablename]
| quit
|
+-----+
SQL>
```

## 4.2.9.9.7 Consumption of the SQL Service for Data Federation

Learn how you can use ODBC-based clients for loading and transforming data.

After exposing an SQL service in the ABAP system and installing the ODBC driver for ABAP, the SQL service and the ODBC driver can be used to provide data access from external ODBC-based clients.

### 4.2.9.9.7.1 Example: Loading and Transforming Data Using Microsoft Excel

After creating a DSN, you can use this DSN in an ODBC client tool of your choice. Learn how you can use Microsoft Excel for loading and transforming data as an example.

#### Procedure

1. Start Microsoft Excel.
2. In the menu, choose  **Data**  **Get Data**  **From Other Sources**  **From ODBC**.
3. In the following popup, enter the DSN.
4. **Communication user only:** In the **Database** section of the popup, you're prompted for user and password to log on to the ABAP system. Enter the user and password that you remember from when you created the communication system (see [Creating a Communication System and a Communication User \[page 1215\]](#)).
5. **Business user only:** In the **Database** section of the popup, you're prompted for user and password. You can enter any user or password; authentication is not checked at this point.
6. Choose **Connect**.

**Business user only:** When a connection is opened by Excel, a browser window with a logon dialog for the ABAP system appears. Depending on your system configuration, the browser logon might ask you for your credentials (user and password). Enter your business user name and password for the ABAP system.

The navigator appears and shows all exposed objects in the SQL schema that you've chosen to expose, that is, in the example used here, the SQL schema `zorders`.

7. To display a preview of the data content in Excel, choose one of the CDS entities.
8. Choose either **Load** to load the data into an Excel sheet or choose **Transform Data** to switch to Power Query in Excel.

When you load the data into an Excel sheet, you can always refresh the data if needed.

9. To execute a free-style SQL query on the exposed entities, choose  **Data**  **Get Data**  **From Other Sources**  **From ODBC** again from the menu in Excel and then choose **Advanced Options**.
10. In the new control, you can enter a **SELECT** statement.

In the **SELECT** statement, you must prefix all view names by your schema name (in the example used here, `zorders`). Apart from these necessary prefixes, you can use an ANSI-like SQL syntax. The result set shows up in an Excel preview window.

## 4.2.9.9.7.2 Example: Loading and Transforming Data Using LibreOffice

After the successful configuration of the ABAP system and the installation of the ODBC driver for ABAP, you can load and transform data from the exposed CDS view entities. You can use external tools such as LibreOffice on Linux, for example.

### Context

#### ⓘ Note

**Relevant for business user access using browser logon:** If you're using unixODBC tools such as `isql` or `iusrsql` on command line as ODBC clients instead of LibreOffice, make sure that you work in an environment such as XWindow where a browser window can open.

### Procedure

1. Start LibreOffice.
2. In LibreOffice Calc, choose **File** **New** **Database**.
3. Choose the *Connect to an existing database* radio button, select *ODBC*, and choose *Next*.
4. Enter the DSN that you inserted into your `.odbc.ini` file and choose *Next*.
5. **Communication user only:** Enter a user name.

**Communication user only:** You can test your connection after providing a password.

6. Choose *Finish*.

**Business user only:** When a connection is opened by LibreOffice, a browser window with a logon dialog for the ABAP system appears. Depending on your system configuration, the browser logon might ask you for your credentials (user and password). Enter your business user name and password for the ABAP system.

LibreOffice now displays all exposed CDS view entities. You can choose these entities to get a data preview.

7. To enter an SQL query string and retrieve the result, choose *Create Query in SQL View*.

#### ⓘ Note

LibreOffice automatically double-quotes all identifiers in the SQL query. Therefore, you must use the mixed case names for the CDS entities that you defined in the ABAP back end.

## 4.2.9.9.8 Consumption of the SQL Service for Data Replication

Learn how you can set up SAP Datasphere as a client for replicating data.

After you have created and exposed an SQL service for data replication (see [Creating a Communication Arrangement for Exposing the SQL Service \[page 1216\]](#)), create a communication arrangement for SAP Datasphere to be able to reach the ABAP system. In addition, create a replication flow in SAP Datasphere to connect the ABAP source system with the envisioned target environment.

After the objects exposed using the SQL service have been selected and deployed in the replication flow, you can access the replicated data in SAP Datasphere for further processing and combining it with data from other data sources.

[Creating a Communication Arrangement to Enable Replication Flows in SAP Datasphere \[page 1229\]](#)

Create a communication arrangement based the integration scenario `SAP_COM_0532` to enable replication flows in SAP Datasphere.

[Creating a Replication Flow for SQL Service Exposure \[page 1230\]](#)

In SAP Datasphere, define a replication flow for CDS view entities using the SQL service exposure.

### 4.2.9.9.8.1 Creating a Communication Arrangement to Enable Replication Flows in SAP Datasphere

Create a communication arrangement based the integration scenario `SAP_COM_0532` to enable replication flows in SAP Datasphere.

#### Prerequisites

You have already created a communication arrangement based on a communication scenario with object privileges so that the communication user in the ABAP system can access the service binding of the SQL service (see [Creating a Communication Scenario with Object Privileges \[page 1214\]](#) and [Creating a Communication Arrangement for Exposing the SQL Service \[page 1216\]](#)).

#### Context

You need the integration scenario `SAP_COM_0532` if you want to use replication flows in SAP Datasphere to subscribe to CDS view entities from an ABAP system and consume data from these views. With a communication arrangement based on `SAP_COM_0532` and with a communication arrangement for exposing the SQL service, you can now consume data from public CDS view entities released by SAP and from **custom** CDS view entities that are exposed using an ABAP SQL service.

## Procedure

1. Create a communication arrangement based on the integration scenario SAP\_COM\_0532.
2. In the communication arrangement, enter the communication system and communication user that you created for the service binding (see *Prerequisites*).

## Related Information

[Integrating SQL Services Using SAP Datasphere \[page 2875\]](#)

### 4.2.9.9.8.2 Creating a Replication Flow for SQL Service Exposure

In SAP Datasphere, define a replication flow for CDS view entities using the SQL service exposure.

## Context

With defining a source of the replication flow, you specify the connection, the folder, and the replication objects (CDS view entities).

For more information about creating replication flows, see [Creating a Replication Flow](#) in the SAP Datasphere documentation.

## Procedure

1. To create a replication flow, follow the instructions of the SAP Datasphere documentation (see [Creating a Replication Flow](#)).
2. As a source connection, choose the communication system that you have created before for connecting SAP Datasphere to your ABAP environment (see also [Creating a Communication Arrangement to Enable Replication Flows in SAP Datasphere \[page 1229\]](#)).
3. As a source container, choose `SQL_SERVICE`.

#### Note

The `SQL_SERVICE` contains the SQL services that have been enabled for replication.

4. As source objects, choose the CDS view entity that you want to use for initial and delta replication.

If an alias was provided, the alias name is shown.

You can choose among the custom CDS view entities that comply to the overall rules for data replication (see [Additional Prerequisites and Constraints for Replication \[page 1207\]](#)). CDS view entities that support delta replication must have been annotated using `@DataIntegration.deltaReplication.intended: true`. All other CDS view entities support only full replication.

5. Choose a target system.
6. As a load type, select *Initial and Delta*.

## 4.2.9.9.9 Developing SQL Queries on CDS Objects Exposed as SQL Service

You can develop your own SQL queries on objects exposed using an ABAP SQL service.

For example, in some cases, you want to use SQL client software applications to interact with the ABAP system using the ODBC driver and using your own defined SQL queries. In other cases, you want to access data using the ODBC driver with programs written in programming languages such as Python or Node.js.

In all these cases, you can access CDS view entities in an ABAP system using SQL via Open Database Connectivity (ODBC). However, you need to know more details on the supported SQL dialect of the ABAP SQL service to be able to define your SQL queries appropriately. This dialect is what is outlined in this documentation.

### ⓘ Note

If you use standard tools with an ODBC client like MS Excel or LibreOffice, the tools themselves assemble the SQL queries executed over the ODBC driver. The tools assume some kind of ANSI SQL compatibility and can derive details about the supported SQL dialect from ODBC `SQLGetInfo()` attributes.

### 4.2.9.9.1 SQL Dialect of the ABAP SQL Service

The SQL dialect of the ABAP SQL service is a modified ABAP SQL dialect executed by the ABAP SQL engine.

#### Basic Features

The SQL dialect of the ABAP SQL service shares many features with ABAP SQL, which is a subset of SQL realized using ABAP statements. In fact, the SQL dialect of the ABAP SQL service is a modified ABAP SQL dialect executed by the ABAP SQL engine. It works on ABAP-managed entities and profits from many ABAP server-specific features like automatic client handling, table buffering, and DCL handling.

However, in some respects, the SQL dialect of the ABAP SQL service is more like ANSI SQL or HANA SQL, the native SQL dialect of the SAP HANA database. For example, in ODBC, there are no ABAP host variables or internal tables, which exist in ABAP SQL. Parameter markers and type casts in the SQL dialect of the ABAP SQL service are handled similar to HANA SQL.

The SQL dialect of the ABAP SQL service is restricted to read-only SQL queries that don't set locks on database level. No `INSERT/UPDATE/DELETE` statements and no `SELECT FOR UPDATE` are allowed on objects

exposed in an SQL service. Therefore, the SQL service has no concept of database transactions and COMMIT or ROLLBACK statements aren't needed.

In addition, there are also some features that are specific to the SQL dialect of the ABAP SQL service and that can't be found in ABAP or HANA SQL. For example, to access objects, a schema name and an object name must be used. What's also new is that the SQL service provides a set of system tables where metadata can be retrieved using SQL.

## Documentation Scope and More Information

This documentation gives you a brief overview of the supported SQL features, highlighting some of the differences and similarities to HANA SQL and ABAP SQL. For more information about HANA SQL and ABAP SQL, see the following documentation:

[ABAP SQL](#)

[SAP HANA SQL Reference Guide for SAP HANA Platform](#)

### 4.2.9.9.9.1.1 Using a Schema Name to Refer to the Service Binding Name

In SQL statements of the SQL dialect of the ABAP SQL service, all accessed objects must be fully qualified using a schema name and an object name. As a schema name, you use the SQL service binding name. The object names are the external names of the exposed CDS objects listed in the corresponding service definition. A simple example is the following SQL statement:

#### ↔ Sample Code

```
Select * from "ZData"."OrderItems"  
where "ZData"."OrderItems"."OrderId" = 42
```

### 4.2.9.9.9.1.2 Prepared Statements with Parameter Markers

Like in HANA SQL, statement texts can contain unnamed parameter markers ( '? ' ).

These statements can be prepared once and executed multiple times using different value bindings at execution time. A simple example is the following SQL statement with four input parameters:

#### ↔ Sample Code

```
Select * from "ZData"."OrderItems"  
where "OrderId" = ? or "OrderId" IN ( ?, ?, ? )
```

The data types of the parameter markers are restricted to simple DDIC/ODBC types and no table-like input is supported.

### 4.2.9.9.9.1.3 Retrieving Table Metadata Using System Tables

In the SQL dialect of the ABAP SQL service, you can use system tables to retrieve table metadata. For example, the system table `SYS.VIEWS` provides data similar to the ODBC function `SQLTables()`, `SYS.COLUMNS` provides data similar to what you get using `SQLColumns()`. With `SYS.VIEW_PARAMETERS`, you can get view parameters.

#### Motivation

In ODBC, object metadata can be retrieved using ODBC functions like `SQLTables()` or `SQLColumns()`. However, in SQL services, view entities with parameters can also be exposed and ODBC has no clear concept for these objects. Therefore, the SQL service also provides a set of system tables where metadata can be retrieved using SQL.

The system tables are automatically exposed and contain information about the objects that can be accessed in the current ODBC connection to the ABAP back end. The objects visible in an ODBC connection depend on the ABAP user privileges and on the service path used to open the ODBC connection (with a communication or a business user).

#### SYS.VIEWS

The system table `SYS.VIEWS` works similar to the ODBC function `SQLTables()` and lists all CDS view entities visible in the current ODBC connection including the system tables themselves. The column `HAS_PARAMETERS` can be used to determine whether a view has parameters:

##### ↔ Sample Code

```
SQL> select schema_name, view_name from SYS.VIEWS where schema_name = 'SYS'  
+-----+-----+  
| SCHEMA_NAME | VIEW_NAME |  
+-----+-----+  
| SYS          | DUMMY      |  
| SYS          | M_CONNECTION_INFO |  
| SYS          | VIEWS      |  
| SYS          | VIEW_COLUMNS |  
| SYS          | VIEW_PARAMETERS |  
+-----+-----+
```

## SYS.VIEW\_COLUMNS

The system table `SYS.VIEW_COLUMNS` lists column information for the exposed entities containing both ODBC and ABAP metadata:

### ↳ Sample Code

```
SQL> select COLUMN_NAME, ODBC_DATA_TYPE OD,
      ODBC_BUFFER_LENGTH OBL, DDIC_TYPE_NAME, DDIC_LENGTH
    from SYS.VIEW_COLUMNS
   where schema_name = 'SYS' and view_name = 'VIEW_COLUMNS'
     order by column_position
```

COLUMN_NAME	OD	OBL	DDIC_TYPE_NAME	DDIC_LENGTH
SCHEMA_NAME	-8	60	CHAR	30
VIEW_NAME	-8	60	CHAR	30
COLUMN_NAME	-8	60	CHAR	30
SCHEMA_NAME_UPPER	-8	60	CHAR	30
VIEW_NAME_UPPER	-8	60	CHAR	30
COLUMN_NAME_UPPER	-8	60	CHAR	30
COLUMN_POSITION	4	4	INT4	10
DESCRIPTION	-8	500	CHAR	250
IS_KEY	-8	10	CHAR	5
ODBC_DATA_TYPE	5	2	INT2	5
ODBC_TYPE_NAME	-8	8	CHAR	4
ODBC_COLUMN_SIZE	4	4	INT4	10
ODBC_DECIMAL_DIGITS	5	2	INT2	5
ODBC_BUFFER_LENGTH	4	4	INT4	10
ODBC_NUM_PREC_RADIX	5	2	INT2	5
ODBC_NULLABLE	5	2	INT2	5
DDIC_TYPE_NAME	-8	8	CHAR	4
DDIC_LENGTH	4	4	INT4	10
DDIC_DECIMALS	5	2	INT2	5

## SYS.VIEW\_PARAMETERS

The system table `SYS.VIEW_PARAMETERS` contains ODBC and ABAP metadata for view parameters (`HAS_PARAMETERS` = 'TRUE' in `SYS.VIEWS`). For an example, see [Named Parameter Syntax on Views with Parameters \[page 1243\]](#).

## Other System and Monitoring Tables

`SYS.DUMMY` is a dummy table with one row and works like the corresponding SAP HANA database table. Many of the SQL examples throughout this documentation use this dummy table.

The monitoring view `SYS.M_CONNECTION_INFO` contains information about the current ODBC connection (for example, ABAP user name, alias, client, SQL service name, and EPP information).

### Note

You can't use the system tables `SYS.VIEWS`, `SYS.VIEW_COLUMNS`, and `SYS.VIEW_PARAMETERS` to retrieve metadata about CDS SQL-based scalar functions. For CDS scalar functions, use the system tables `SYS.FUNCTIONS` and `SYS.FUNCTION_PARAMETERS`.

## Related Information

[ABAP CDS: SQL-Based Scalar Functions \[page 1241\]](#)

### 4.2.9.9.1.4 Set Operations UNION, INTERSECT, EXCEPT

Like in ABAP SQL and HANA SQL, you can use the standard set operations `UNION`, `INTERSECT`, and `EXCEPT` to combine result sets of multiple queries in one query.

Here are some examples:

#### Sample Code

```
select DUMMY FROM SYS.DUMMY A EXCEPT SELECT 'X' FROM SYS.DUMMY B  
select DUMMY FROM SYS.DUMMY EXCEPT SELECT 'X' FROM SYS.DUMMY
```

In cases where the same table name is used in multiple `UNION ALL` branches, the ABAP SQL engine requires that these tables are given different alias names.

### 4.2.9.9.1.5 Supported JOIN Syntax

The supported `JOIN` syntax in the SQL dialect of the ABAP SQL service is similar to HANA SQL.

The commonly known ANSI SQL clauses `INNER JOIN`, `LEFT OUTER JOIN`, `RIGHT OUTER JOIN`, and `CROSS JOIN` are supported. For a `LEFT OUTER JOIN`, you can also add the SAP HANA-specific `LEFT OUTER MANY TO ONE JOIN` clause.

Instead of using explicit `JOIN` clauses, you can also use a comma-separated list of tables in the `FROM` clause to express an `INNER JOIN` or a `CROSS JOIN`. This feature isn't supported in ABAP SQL, but in the SQL dialect of the ABAP SQL service it is.

Here are some examples:

#### Sample Code

```
select A.DUMMY FROM SYS.DUMMY A  
CROSS JOIN SYS.DUMMY B  
select A.DUMMY FROM SYS.DUMMY A , SYS.DUMMY B  
select A.DUMMY FROM SYS.DUMMY A  
LEFT OUTER MANY TO ONE JOIN SYS.DUMMY B
```

```
ON A.DUMMY = B.DUMMY
```

## 4.2.9.9.9.1.6 Common Table Expressions and Derived Tables

Common table expressions (CTE) can be used to define temporary view objects using a `WITH` clause at the beginning of a query that can later be used multiple times in the full select part.

Here's an example:

### ↳ Sample Code

```
WITH A AS ( SELECT SCHEMA_NAME S, VIEW_NAME V FROM SYS.VIEWS )
SELECT * FROM A WHERE A.S = 'SYS'
UNION ALL
SELECT * FROM A WHERE A.S = 'ZData'
```

Recursive SQL in CTEs isn't supported.

As an alternative to CTEs, derived table clauses can be directly used in the `FROM` clause of a query, as in the following example:

### ↳ Sample Code

```
SELECT * FROM ( SELECT SCHEMA_NAME S, VIEW_NAME V FROM SYS.VIEWS ) A
WHERE A.S = 'SYS'
```

While CTEs are also supported in ABAP SQL, derived table clauses are only supported in the SQL dialect of the ABAP SQL service.

Both queries with CTEs and with derived tables work a little different regarding case sensitivity of object names. Whenever you define a name in double quotes for the CTE or derived table itself or for returned column names, you must use the name in double quotes throughout the whole SQL statement, as in the following example:

### ↳ Sample Code

```
SELECT * FROM ( SELECT SCHEMA_NAME "s", VIEW_NAME V FROM SYS.VIEWS ) "a"
WHERE "a"."s" = 'SYS'
```

The following query doesn't work because the double quotes are not used consistently:

### ↳ Sample Code

```
SELECT * FROM ( SELECT SCHEMA_NAME "s", VIEW_NAME V FROM SYS.VIEWS ) "a"
WHERE A.S = 'SYS'
```

## 4.2.9.9.1.7 Type Casts

You can also use type casts in the SQL dialect of the ABAP SQL service.

You can always insert type casts in ABAP CDS style directly in the definition of your exposed CDS entities. However, sometimes you may want to insert type casts in your SQL queries, for example, to make some data types better consumable or readable in standard ODBC applications.

The supported type casts in the SQL dialect of the ABAP SQL service are more HANA SQL-like. It's only possible to cast to a limited number of ABAP types. The general form of a type cast is the following, where <expression> can be an SQL expression, a literal value, or a column:

### ↔ Sample Code

```
CAST( <expression> AS <data type> )
```

<data type> can be one of the data types from the second column of the following table:

	Data Type in Cast	Corresponding ABAP Data Type
Integer data types	SMALLINT, INTEGER, BIGINT	abap.int2, abap.int4, abap.int8
Fixed-point decimal types	DECIMAL( p , s ) where p <= 31 and s <= 14	abap.dec(p,s)
Double	DOUBLE	abap.double
Character-like types	CHAR( n ) where n <= 255  Synonyms: VARCHAR( n ), NVARCHAR( n )	abap.char(n)
Decimal floating point types	DECIMAL, SMALLDECIMAL	abap.df34n, abap.df16n

A character string can be cast to an abap.string using the TO\_CLOB( <expression> ) function. A binary string can be cast to an abap.xstring using the TO\_BLOB( <expression> ) function. There are no special casts to date, time, or timestamp data types and also no type casts for some of the ABAP-specific data types like abap.numc, abap.curr, abap.cuky, and so on.

Here's an example:

### ↔ Sample Code

```
select      CAST( 4711 AS SMALLINT ) AS C_INT2,  
            CAST( 4711 AS INTEGER ) AS C_INT4,  
            CAST( 1E-1200 AS DECIMAL) AS C_DF34N,  
            CAST( 1.234 AS DECIMAL(10,3)) AS C_DEC103,  
            CAST( 2.5 AS DOUBLE) AS C_DOUBLE  
FROM SYS.DUMMY
```

## 4.2.9.9.1.8 Literals and ODBC Escape Sequences

Literal values can be used in the SQL dialect of the ABAP SQL service similar to HANA SQL.

For numeric literals, you can add an explicit type cast to enforce a specific data type in the result set. For character-like data types, you can enclose the literals in single quotes. A prefix `N` is optional. Raw (binary) literals must be enclosed in single quotes with prefix `x`. In the single quotes, you provide the hex string representation of the raw value. For DATE/TIME/TIMESTAMP (`abap.datn/abap.timn/abap.utcl`) literals, the prefixes date, time, or timestamp can be used.

Here's an example:

### ↔ Sample Code

```
SQL> SELECT 'HELLO' AS C_CHAR5,
      x'123456ABCDEF' AS C_RAW6,
      date'2015-01-02' AS C_DATN,
      time'11:55:00' AS C_TIMN,
      timestamp '2015-01-02 11:55:00.1234567' AS C_UTCL
FROM SYS.DUMMY
+-----+-----+-----+-----+
| C_CHAR5 | C_RAW6      | C_DATN   | C_TIMN   |
+-----+-----+-----+-----+
| HELLO   | 123456ABCDEF | 20150102 | 115500   | 2015-01-02T11:55:00,1234567 |
+-----+-----+-----+-----+
```

As an alternative for DATE/TIME/TIMESTAMP, you can use ODBC escape sequences, which are also supported in HANA SQL:

### ↔ Sample Code

```
SQL> SELECT {d '2015-01-02'} AS C_DATN,
      {t '11:55:00'} AS C_TIMN,
      {ts '2015-01-02 11:55:00.1234567'} AS C_UTCL
FROM SYS.DUMMY
+-----+-----+-----+
| C_DATN | C_TIMN   | C_UTCL   |
+-----+-----+-----+
| 20150102 | 115500   | 2015-01-02T11:55:00,1234567 |
+-----+-----+-----+
```

## 4.2.9.9.1.9 Built-In SQL Functions

Most of the supported built-in scalar and aggregate ABAP SQL functions can be used, but there are some limitations.

### General Comments

#### ⓘ Note

This documentation is about built-in SQL functions. For more information about CDS SQL-based scalar functions, see [ABAP CDS: SQL-Based Scalar Functions \[page 1241\]](#).

Since the SQL dialect of the ABAP SQL service is eventually processed by the ABAP SQL engine, most of the supported built-in scalar ABAP SQL functions can be used.

In some of the date/time functions, ABAP data type names are reflected in the function names, as in the following example:

#### ↗ Sample Code

```
SQL> SELECT date'2015-01-02'      AS C_DATN,
          ADD_DAYS( date'2015-01-02', 5 )      AS C_DATN_5,
          DATN_ADD_DAYS( date'2015-01-02', 10 ) AS C_DATN_10
     FROM SYS.DUMMY
+-----+-----+-----+
| C_DATN | C_DATN_5 | C_DATN_10 |
+-----+-----+-----+
| 2015-01-02 | 2015-01-07 | 2015-01-12 |
+-----+-----+-----+
```

The functions DATS\_FROM\_DATN, DATS\_TO\_DATN, BINTOHEX, and HEXTOBIN can also be used for type casting.

Apart from scalar functions, ABAP SQL aggregate functions and ABAP SQL windowing functions can also be used in the SQL dialect of the ABAP SQL service.

It's not possible to access associations using path expressions in the SQL dialect of the ABAP SQL service. Columns of associated entities can only be accessed if they're explicitly exposed using a column name in the CDS view entity.

## Limitations for Built-In Functions

For the SQL dialect of the ABAP SQL service, the following limitations apply:

String Functions

Function	Supported	Comments
LIKE_REGEXPR	Yes	Only supported as comparison operator, the <code>case_sensitive</code> parameter is not supported.  Different syntax: <code>LIKE_REGEXPR( pcre IN value [FROM start] [OCCURRENCE occ] [GROUP group] )</code>
LOCATE_REGEXPR	Yes	The <code>case_sensitive</code> parameter is not supported.  Different syntax: <code>LOCATE_REGEXPR( pcre IN value [FROM start] [OCCURRENCE occ] [GROUP group] )</code>
LOCATE_REGEXPR_AFTER	Yes	The <code>case_sensitive</code> parameter is not supported.  Different syntax: <code>LOCATE_REGEXPR( AFTER pcre IN value [FROM start] [OCCURRENCE occ] [GROUP group] )</code>
OCCURENCES_REGEXPR	Yes	The <code>case_sensitive</code> parameter is not supported.  Different syntax: <code>OCCURENCES_REGEXPR( pcre IN value )</code>
REPLACE_REGEXPR	Yes	The <code>case_sensitive</code> parameter is not supported.  Different syntax: <code>REPLACE_REGEXPR( pcre IN sql_exp1 WITH sql_exp2 [FROM start] [OCCURRENCE occ] )</code>

Function	Supported	Comments
SUBSTRING_REGEXPR	Yes	<p>The <code>case_sensitive</code> parameter is not supported.</p> <p>Different syntax:</p> <pre>SUBSTRING_REGEXPR( pcre IN value [FROM start] [ OCCURRENCE occ ] [ GROUP group ] )</pre>

#### Time Zone Functions

Function	Supported	Comments
ABAP_SYSTEM_TIMEZONE	Yes	Only with default parameters, named parameters are not supported.
ABAP_USER_TIMEZONE	Yes	Only with default parameters, named parameters are not supported.

The following conversion functions are not supported:

- `UNIT_CONVERSION`
- `CURRENCY_CONVERSION`

## More Information

For more information about the ABAP SQL functions, see the ABAP SQL documentation.

## 4.2.9.9.1.10 ABAP CDS: SQL-Based Scalar Functions

In the SQL dialect of the ABAP SQL service, you can use the system views `SYS.FUNCTIONS` and `SYS.FUNCTION_PARAMETERS` to discover CDS SQL-based scalar functions and retrieve their metadata.

## Motivation and Background

When you are using SQL services to access ABAP-managed data, you might want to use CDS SQL-based scalar functions. SQL-based scalar functions are scalar functions that are evaluated by an SQL environment.

However, the CDS scalar functions don't have a representation in the metadata access functions of ODBC. The CDS scalar functions are **not** represented in the system views `SYS.VIEWS` and `SYS.VIEW_PARAMETERS`, either, which are used for metadata retrieval for the SQL service (see [Retrieving Table Metadata Using System Tables \[page 1233\]](#)).

Therefore, for the SQL service, the system views `SYS.FUNCTIONS` and `SYS.FUNCTION_PARAMETERS` are available so that metadata of CDS SQL-based scalar functions can be exposed.

## Metadata Views `SYS.FUNCTIONS` and `SYS.FUNCTION_PARAMETERS`

The system view `SYS.FUNCTIONS` exposes the function name and description of all SQL-based scalar functions for which the flag *Auto Exposure* has been set. This flag applies to all SQL-based scalar functions released by SAP for auto exposure; you can also set it for your own custom SQL-based scalar functions. The system view `SYS.FUNCTION_PARAMETERS` exposes detail information about the parameters of the SQL-based scalar functions.

The system views `SYS.FUNCTIONS` and `SYS.FUNCTION_PARAMETERS` are automatically exposed with the CDS name and can be used in requests to the SQL service without any restrictions. No additional authorizations and no further explicit declarations are needed. The system views are intended for scalar functions for specialized conversions or system-wide valid common expressions, for example, without exposing application data.

The system views are exposed with their CDS name with mixed case; there is no possibility to define an alias name. They don't belong to a schema in the SQL service, but they are accessed like built-in CDS scalar functions (for example, `CONCAT`) without a schema qualification.

## Constraints

With the SQL service, you can't use scalar functions that have parameters of a generic ABAP type such as `NUMERIC` or that use reference types.

## Code Samples

Using this code sample, you can discover available CDS SQL-based scalar functions that are automatically exposed in your system:

### ↔ Sample Code

```
SQL> select FUNCTION_NAME, DESCRIPTION
   FROM SYS.FUNCTIONS S
  WHERE S.FUNCTION_TYPE = 'SCALAR'
    AND S.SCHEMA_NAME    = ''
```

### ⓘ Note

To find automatically exposed functions (which must be accessed without schema name qualification), you must use the filter `SCHEMA_NAME = ''`. To make sure that you only get the scalar functions, use the filter `FUNCTION_TYPE = 'SCALAR'`.

The following code sample illustrates how you can get information about the parameters of the demo scalar function DEMO\_CDS\_DECFLOAT\_RATIO:

#### ↔ Sample Code

```
SQL> select PARAMETER_NAME, PARAMETER_TYPE, DDIC_TYPE_NAME, ODBC_DATA_TYPE,
      ODBC_COLUMN_SIZE
      FROM SYS.FUNCTIONS S
      INNER JOIN SYS.FUNCTION_PARAMETERS P ON S.SCHEMA_NAME = P.SCHEMA_NAME and
      S.FUNCTION_NAME = P.FUNCTION_NAME
      WHERE S.FUNCTION_TYPE = 'SCALAR'
      AND S.SCHEMA_NAME    = ''
      AND S.FUNCTION_NAME  = 'DEMO_CDS_DECFLOAT_RATIO'
      +-----+-----+
      +-----+-----+
      | PARAMETER_NAME          | PARAMETER_TYPE | DDIC_TYPE_NAME |
      ODBC_DATA_TYPE | ODBC_COLUMN_SIZE |
      +-----+-----+
      +-----+-----+
      | base                  | IN           | D34N          |
      -360          | 34          | IN           | D34N          |
      | portion                | RETURN       | D34N          |
      -360          | 34          |              |              |
      +-----+-----+
      +-----+-----+
      SQLRowCount returns 0
      3 rows fetched
```

In the following code sample, the demo scalar function DEMO\_CDS\_DECFLOAT\_RATIO is applied to calculate a ratio:

#### ↔ Sample Code

```
SQL> select DEMO_CDS_DECFLOAT_RATIO( base => CAST( 200 AS DECIMAL) , portion
      => CAST( 50 AS DECIMAL) ) FROM SYS.DUMMY
      +-----+
      |
      +-----+
      | 25
      +-----+
      SQLRowCount returns 0
      1 rows fetched
```

## 4.2.9.9.1.11 Named Parameter Syntax on Views with Parameters

The SQL dialect of the ABAP SQL service supports named parameter syntax on views with parameters.

#### ⓘ Note

The named parameter syntax is also supported in HANA SQL. However, the SQL dialect of the ABAP SQL service doesn't support positioned parameter syntax.

Here's a simple example with a CDS entity with one parameter to illustrate how it works:

#### ↳ Sample Code

```
@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_ALLOWED
@EndUserText.label: 'Test entity with parameter'
define view entity ZENTITYWITHPARAM with parameters
  im_int: abap.int4
as select from svers {
  key version as keyCol,
  $parameters.im_int as IntCol
}
```

Let's also assume you've exposed this entity in your service definition with external name `ZEntityWithParam` and with corresponding service binding `ZData`.

After activating all ABAP objects, the system views will now show the following entries:

#### ↳ Sample Code

```
SQL> select HAS_PARAMETERS from SYS.VIEWS
      WHERE schema_name = 'ZData'
            and view_name    = 'ZEntityWithParam'
+-----+
| HAS_PARAMETERS |
+-----+
| TRUE           |
+-----+
SQL> select PARAMETER_NAME, ODBC_DATA_TYPE, DDIC_TYPE_NAME, DDIC_LENGTH
      from SYS.VIEW_PARAMETERS
      WHERE schema_name = 'ZData'
            and view_name    = 'ZEntityWithParam'
+-----+-----+-----+-----+
| PARAMETER_NAME | ODBC_DATA_TYPE | DDIC_TYPE_NAME | DDIC_LENGTH |
+-----+-----+-----+-----+
| im_int        | 4             | INT4          | 4           |
+-----+-----+-----+-----+
SQL> select COLUMN_NAME, ODBC_DATA_TYPE, DDIC_TYPE_NAME, DDIC_LENGTH
      from SYS.VIEW_COLUMNS
      WHERE schema_name = 'ZData'
            and view_name    = 'ZEntityWithParam'
+-----+-----+-----+-----+
| COLUMN_NAME   | ODBC_DATA_TYPE | DDIC_TYPE_NAME | DDIC_LENGTH |
+-----+-----+-----+-----+
| keyCol        | -8            | CHAR          | 72          |
| IntCol        | 4             | INT4          | 4           |
+-----+-----+-----+-----+
```

Now, you can run SQL queries on the new entity with named parameter syntax:

#### ↳ Sample Code

```
SQL> select * from ZData.ZEntityWithParam ( im_int => 4711 )
+-----+
| keyCol     | IntCol    |
+-----+
| 7xx        | 4711      |
+-----+
```

## 4.2.9.9.1.12 ABAP to ODBC Data Type Mappings

The ODBC driver for ABAP allows you to configure the mapping of ABAP data types to ODBC SQL data types.

To configure this mapping, you can use the connection parameter `Typemap`. Regardless of the ODBC SQL data type reported by the ODBC driver, the ODBC application is still free to use an ODBC C data type of its choice to retrieve a given ODBC SQL data type.

### Typemap=semantic

In most cases, you want to retrieve the ABAP data in a way that's easy to consume. With the `Typemap=semantic` configuration, the ODBC driver can do the following:

- Map all date like data types like `abap.dat`s and `abap.datn` to ODBC type `SQL_DATE`/ `SQL_TYPE_DATE`
- Map all time like data types like `abap.ti`m`s` and `abap.tim`n to ODBC type `SQL_DATE`/ `SQL_TYPE_DATE`
- Map all decimal floating point-like data types like `abap.d16n`, `abap.d34n`, `abap.d16r`, `abap.d34r`, `abap.d16d`, and `abap.d34d` to ODBC type `SQL_DECFLOAT`
- Perform an automatic decimal shift for type `abap.curr` and maps this data type also to ODBC type `SQL_DECFLOAT`
- Perform an ISO conversion for type `abap.lang` and maps this data type to ODBC type `SQL_WCHAR` with length 2

### Typemap=semanticDatsTimsAsWchar

For some ODBC applications that want to safely access the ABAP data, the `Typemap=semantic` configuration might have the problem that not all data in the old `abap.dat`s and `abap.ti`m`s` columns can be mapped to a date. The ABAP system doesn't prevent that some applications store date/time literals in those columns (for example, the string `TODAY` in an `abap.dat`s column).

To ensure that these use cases work without data loss, a slightly different `Typemap=semanticDatsTimsAsWchar` is available. This type mapping is similar to the semantic type mapping, with the exception that the ABAP data type `abap.dat`s is mapped to `SQL_WCHAR` with length 8 and ABAP data type `abap.ti`m`s` is mapped to `SQL_WCHAR` with length 6. As a result, no conversions happen, and no potential conversion errors occur.

### Typemap=native

Other applications might want to read data from an ABAP system as-is without any semantic conversion. With the `Typemap=native` configuration, the ODBC driver does the following:

- Map data type `abap.dat`s to ODBC type `SQL_WCHAR` with length 8
- Map data type `abap.datn` to ODBC type `SQL_DATE`/ `SQL_TYPE_DATE`

- Map data type `abap.tims` to ODBC type `SQL_WCHAR` with length 6
- Map data type `abap.timn` to ODBC type `SQL_TIME/SQL_TYPE_TIME`
- Map data types `abap.d16n` and `abap.d34n` to ODBC type `SQL_DECFLOAT`
- Map data types `abap.d16r` and `abap.d34r` to ODBC type `SQL_BINARY`
- Map data types `abap.d16d` and `abap.d34d` to ODBC type `SQL_DECIMAL`
- Not shift `abap.curr` data and maps of this type to ODBC type `SQL_DECIMAL`
- Return ABAP type `abap.lang` as is and map it to ODBC type `SQL_WCHAR` with length 1

All other ABAP data types are mapped to their corresponding ODBC SQL types independent of the typemap setting, such as the following, for example:

ABAP Data Type	Mapped ODBC SQL Type
<code>INT1/INT2</code>	<code>SQL_SMALLINT</code>
<code>INT4</code>	<code>SQL_INTEGER</code>
<code>INT8</code>	<code>SQL_BIGINT</code>
<code>DEC</code>	<code>SQL_NUMERIC</code>
<code>UTCL</code>	<code>SQL_TIMESTAMP/SQL_TYPE_TIMESTAMP</code>

## 4.2.9.9.1.13 Data Conversions for `abap.lang` and `abap.curr` Data Types

There's a special handling of ABAP data types `LANG` and `CURR` in the ODBC driver.

### abap.lang Data Type

In ABAP tables, `abap.lang` data is stored as single-byte Unicode character (for example, `D` for German or `E` for English). However, in external interfaces (for example, the SAP logon screen), the language is typically shown as two-character string (for example, `DE` for German or `EN` for English). The table `T002` in the ABAP back-end system defines the mapping between these values.

If you're using one of the `Typemap=semantic*` settings, `abap.lang` values in the result set of a query are automatically mapped to the 2-character strings. Conversely, if you're using `abap.lang` values as input parameters, you must provide them as 2-character strings. The same SQL queries return different `abap.lang` values and need different `abap.lang` input values if `Typemap=native` is used.

### abap.curr Data Type

The `abap.curr` data type has been developed in times where no decimal floating point data type existed. For historic reasons, on database level, `abap.curr` data is stored in fixed-point decimal values. These values can only be interpreted correctly if the corresponding `abap.cuky` value is known. The fixed-point decimal

values need to be shifted depending on its currency key. For an innocent or generic ODBC application that doesn't know the ABAP shifting algorithm, it comes handy that the ODBC driver for ABAP directly returns the shifted `abap.curr` values. As a price, a fixed-point decimal data type can no longer hold values for all possible currencies. Therefore, the ODBC driver for ABAP returns shifted `abap.curr` values as decimal floating point (ODBC data type `SQL_DECFLOAT`) if one of the `Typemap=semantic*` settings is used.

The following example uses `Typemap=semantic`:

#### ↳ Sample Code

```
SQL> SELECT LANG, CUKY, CURR from ZData.TestView
+-----+-----+
| LANG | CUKY | CURR
+-----+-----+
| EN   | USD  | 9999999999999.99
| DE   | EUR  | 0.01
| LT   | BHD  | 1234.5
+-----+-----+
```

The following example uses `Typemap=native`:

#### ↳ Sample Code

```
SQL> SELECT LANG, CUKY, CURR from ZData.TestView
+-----+-----+
| LANG | CUKY | CURR
+-----+-----+
| E    | USD  | 9999999999999.99
| D    | EUR  | 0.01
| X    | BHD  | 12345.00
+-----+-----+
```

The examples here demonstrate different query results with and without active language conversion and currency shift. Note the changed values for column `LANG`, the change in data type for column `CURR`, and the shifted decimal point in the currency value for currency `BHD`. Currency values for currencies `USD` and `EUR` are stored unshifted in the ABAP database tables.

It's also possible to use shifted `abap.curr` values as input values for SQL queries if they correspond to predicates on a CDS entity column. However, you can't use shifted `abap.curr` values as input parameters for CDS entities with parameters. In this case, the corresponding `abap.cuky` value can't be automatically determined.

## 4.2.9.9.1.14 The Driver-Specific SQL\_DECFLOAT Type

For the decimal floating-point types, the ODBC driver for ABAP has its own ODBC SQL data type `SQL_DECFLOAT=-360` defined.

### Motivation

The ODBC standard doesn't define a decimal floating-point type. However, there are databases like the SAP HANA database or the ABAP system itself (if it's viewed as a database) that support such types. The main

reason for using these types is to avoid rounding problems that would occur when using the double floating-point type with radix 2. ABAP applications often use decimal floating-point data for business data where such rounding could cause problems.

Note the rounding artifacts in the following query result:

#### ↳ Sample Code

```
SQL> select cast (1.05 as double), CAST(1.05 AS DECIMAL) from SYS.DUMMY
+-----+
+-----+
| |
+-----+
+-----+
| 1.050000000000000444089209850063 |
1.05
+-----+
+-----+
SQLRowCount returns 0
1 rows fetched
```

#### ODBC SQL Data Type `SQL_DECFLOAT=-360`

The ODBC standard allows that ODBC drivers define their own driver-specific data types if needed. The ODBC driver for ABAP has taken this way for the decimal floating-point types: It has defined its own ODBC SQL data type `SQL_DECFLOAT=-360` with two different octet lengths corresponding to the `abap.d16n` and `abap.d34n` data types. ODBC applications that want to retrieve such `SQL_DECFLOAT` data must use default target data type `SQL_C_CHAR` or `SQL_C_WCHAR` to return the data in readable string format.

### 4.2.9.9.1.15 Client Handling and Table Buffering

The SQL dialect of the ABAP SQL service is processed by the ABAP SQL engine. As a result, ABAP session variables like the session client are automatically set. On CDS view entities, this ensures that only data from the logon client of the current user is visible. The ABAP SQL engine can also use other ABAP application server-specific mechanisms like table buffering and not all SQL statements need to be routed to the underlying SAP HANA database.

## 4.2.9.10 HTTP Request Processing Constraints

In the SAP BTP, ABAP environment, HTTP request processing has constraints: 600-second timeout and 400 MB max request size.

### Processing Timeout

The processing timeout for HTTP requests in the SAP BTP, ABAP environment is 600 seconds. After this time period, requests are aborted.

 Note

Request processing usually only takes a short amount of time during development because just small data sets are used. Make sure to keep the processing time also for production data set sizes.

### Maximum Request Size

The maximum allowable request size for HTTP requests in the SAP BTP, ABAP environment is 400 MB. Requests that exceed this size limit are rejected to prevent excessive resource consumption and potential performance degradation.

## 4.2.9.11 Integrating Enterprise Event Enablement

In the context of business applications, an event represents a significant change of state that is relevant for follow-up processes. For example, when a new sales order is created, you can use events to trigger additional workflows in other applications.

Event-driven architecture enables asynchronous communication between an event provider and an event consumer in use cases where no direct response from the event consumer is required.

For more information on how to use business events, refer to the [ABAP Integration and Connectivity](#) guide, under [Event-Based Integration](#).

## 4.2.9.11.1 Communication Management

The *Communication Management* app helps you to integrate your SAP BTP, ABAP environment system with SAP BTP, Cloud Foundry environment systems to enable data exchange.

You can use the `SAP_COM_0092` and `SAP_COM_0492` communication scenario IDs to configure a connection with the SAP Event Mesh service to enable the exchange of events. To set up a communication scenario using the *Communication Management* app, follow the steps in the following topics:

- [Creating Technical Communication User \[page 1250\]](#)

To set up the communication arrangement, refer to the respective topics in:

- [Integration with SAP Enterprise Messaging 1.0 for Cloud Foundry \[page 1256\]](#)
- [Integration with Advanced Mesh Service Plan for Service Cloud \[page 1262\]](#)

### 4.2.9.11.1.1 Creating Technical Communication User

You use this procedure to create a communication user for inbound communication from SAP BTP, ABAP environment to SAP S/4HANA Cloud systems.

#### Prerequisites

- You have a subaccount in SAP BTP, Cloud Foundry environment. Refer to [Getting Started with a Trial Account in the Cloud Foundry Environment](#).
- You have created a service instance for SAP Event Mesh or SAP Advanced Mesh Service Plan.
- The key user must have the business role `SAP_BR_ADMINISTRATOR` (Administrator) that contains the business catalog `SAP_CORE_BC_COM` (Communication Management).

#### Context

A communication user enables the integration with other solutions.

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, choose the *Maintain Communication Users* artifact.
3. Choose *New* to create a new user.
4. Enter a *User Name* for the user.

5. Enter a *Description* for the user.
6. Assign a *Password* for the user.
7. Choose *Create*.
8. Make a note of the *User Name*.

The user name is required when you create the communication arrangement.

## Results

You have created a communication user. It is listed in the [Maintain Communication Users](#) artifact.

### 4.2.9.11.2 CloudEvents Context Attributes

Events published through the enterprise event enablement are compliant with the CloudEvents specification. Context attributes contained in events are listed in this topic.

The following table lists required and optional context attributes of the CloudEvents specification (<https://cloudevents.io> ).

Context Attribute	Requirement	Description
id	required	identifies the event instance

Context Attribute	Requirement	Description
source	required	<p>identifies the issuer of the event</p> <p>containing:</p> <ul style="list-style-type: none"> <li>• region           <ul style="list-style-type: none"> <li>• the region where the application service is located</li> <li>• value: default</li> </ul> </li> <li>• applicationNamespace           <ul style="list-style-type: none"> <li>• the registered namespace of the application, which emits the event</li> <li>• value</li> </ul> </li> <li>• sap.s4.beh for SAP released events</li> <li>• sap.abap.custom for events created by customers</li> <li>• instanceID           <ul style="list-style-type: none"> <li>• identifier of the application instance issuing the event</li> <li>• value: represented through the cloud landscape directory (CLD) tenant ID of the ABAP environment tenant.</li> </ul> </li> </ul>
<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p><b> ⓘ Note</b></p> <p>All contained values are set for events published in the ABAP environment.</p> </div>		
specversion	required	<p>specifies the version of the CloudEvents specification, which the event uses</p> <p>currently: 1.0</p>
type	required	specifies the type of event
datacontenttype	optional	content type of the event
time	optional	timestamp of the occurrence of the event
data	optional	business payload of the event

### 4.2.9.11.3 Event Consumer

In this topic, you can find information about how to enable the event consumption for an event consumption model defined using the ABAP Development Tools and how to set up the corresponding communication arrangement.

The communication arrangements SAP\_COM\_0092 or SAP\_COM\_0492 define the connection from ABAP environment to the respective event exchange infrastructure. Via these communication arrangements, the ABAP environment receives events from the event queue and transfers the events to the event consumption model, which then processes the event. The event is processed according to the logic implemented in the method for handling the event in the consumer extension class.

The following steps are required to configure the event consumer:

[Creating a Communication Arrangement for the Communication Scenario bound to an Event Consumption Model \[page 1253\]](#)

To be able to receive events from the event exchange infrastructure service instance, you must configure the inbound communication from the event exchange infrastructure to the consuming event consumption model.

[Checking Channel Binding \[page 1255\]](#)

This topic describes how to check the channel binding for the enterprise event enablement.

## Related Information

[Queue Subscriptions \[page 1261\]](#)

### 4.2.9.11.3.1 Creating a Communication Arrangement for the Communication Scenario bound to an Event Consumption Model

To be able to receive events from the event exchange infrastructure service instance, you must configure the inbound communication from the event exchange infrastructure to the consuming event consumption model.

## Prerequisites

- You've set up a communication arrangement for SAP\_COM\_0092 or SAP\_COM\_0492 as described in [Communication Management \[page 1250\]](#).
- You've created an event consumption model and a *Communication Scenario* as described in [Event Consumption \[page 1279\]](#).

## Context

Following the procedure, you create an inbound communication arrangement.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Communication Arrangements* artifact.

3. Choose *New*.
4. Select the *Communication Scenario* that was created for the respective event consumption model.

You can recognize the scenario by its *Inbound Service*. An *Inbound Service* is created for each event consumption model and later on assigned to the respective communication scenario.

5. Choose an *Arrangement Name* and click on *Create*.
6. To assign a communication system, proceed as follows:

- Assign an existing communication system by selecting a *Communication System*. You can ignore the substeps.
- Create a communication system as described in [How to Create Communication Systems](#).

- a. In the *General* tab under *Technical Data*, pull the switch for *Event Mesh* to *ON*.

Note that communication arrangements created for an event consumption model only support inbound communication. That's why the *Inbound Only* flag is activated when switching *Event Mesh* on.

- b. In the *General* tab under *Technical Data*, select the *Event Channel* that you've configured as described in [Communication Management \[page 1250\]](#).

The *Event Channel* name you see in the value help is the name that was assigned in the property *Channel* of the respective SAP\_COM\_0092 or SAP\_COM\_0492 communication arrangement.

- c. In the *Users for Inbound Communication* tab, add an inbound user.

#### ⓘ Note

The *User for Inbound Communication* (inbound user) and the *Communication User* must be maintained as two different users. Make sure to create a dedicated inbound user, apart from your communication user. To create a new inbound user, follow the steps described in [How to Create Communication Users](#).

The inbound user is used later for event processing. This user has authorizations maintained in the *Authorization Default Values* as described in [Adjusting the Generated Artifacts](#).

- d. Choose *Save*.

In general, you can assign multiple event consumption models via the corresponding communication scenarios to the same SAP\_COM\_0092 or SAP\_COM\_0492 communication arrangement.

## Results

You've created an inbound communication arrangement for your event consumption model.

## Related Information

[How to Create a Communication Arrangement](#)

## 4.2.9.11.3.2 Checking Channel Binding

This topic describes how to check the channel binding for the enterprise event enablement.

### Prerequisites

- You've set up a communication arrangement for SAP\_COM\_0092 or SAP\_COM\_0492 as described in [Communication Management \[page 1250\]](#).
- You've created an event consumption model and a communication scenario as described in [Event Consumption \[page 1279\]](#).

### Context

Inbound topic bindings are automatically created during generation and configuration of an event consumption model by assigning the communication system of your event consumption model to the corresponding communication arrangement. Maintaining inbound topic bindings can only be done through changes of the communication system of your event consumption model.

Follow the steps below to view the inbound topic bindings.

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the [Communication Management](#) app, select the [Enterprise Event Enablement – Configure Channel Binding](#) app.
3. Select the [Channel](#) name of the corresponding communication arrangement to which the communication system of your event consumption model has been assigned to in [Creating a Communication Arrangement for the Communication Scenario bound to an Event Consumption Model \[page 1253\]](#).

In the [Inbound Topic Bindings](#) tab under [Inbound Topics](#), the event topics of the event consumption model are displayed. The topics you see here correspond to the event types you've selected during the creation of the event consumption model. Since these are created automatically, the [Maintained by](#) field has the value [Communication Management](#).

4. Maintain the subscriptions in the [Subscriptions](#) tab as described in [Queue Subscriptions \[page 1261\]](#).

Here you can add all subscription addresses of the respective event exchange infrastructure queues from which you want to consume events.

If a subscription to a queue of the respective event exchange infrastructure service instance is maintained and acknowledged, all events on this queue are consumed by the ABAP environment. Only if the event topics are maintained under [Inbound Topics](#), the incoming events are handed over to the consumer extension class of the corresponding event consumption model. If the event topic isn't maintained, the event is consumed without any further processing.

## 4.2.9.11.4 Integration with SAP Enterprise Messaging 1.0 for Cloud Foundry

This section describes the integration of the enterprise event enablement with the integration scenario SAP\_COM\_0092.

### ⓘ Note

The SAP Enterprise Messaging 1.0 for Cloud Foundry is also referred to as SAP Event Mesh.

### 4.2.9.11.4.1 Communication Arrangements

A communication arrangement is required for the event exchange between ABAP environment and SAP BTP, Cloud Foundry environment.

To use a communication arrangement, you need to create a service instance for the SAP Event Mesh. The service instance contains all information required to establish a connection. This required information, such as endpoints and credentials, is stored in the service key. The enterprise event enablement extracts the information of the service key and creates the corresponding destination and OAuth 2.0 client configuration automatically.

#### 4.2.9.11.4.1.1 Creating Communication Arrangements

In this topic, a communication arrangement is created.

### Prerequisites

- You have a subaccount in the SAP BTP. Refer to [Getting Started](#).
- You have created a service instance for SAP Event Mesh. If you haven't created a service instance before, refer to [Creating an Enterprise Messaging Service Instance](#).

### ⓘ Note

When defining the *Service Descriptor* for the service instance of the SAP Event Mesh service, make sure that the length of the *namespace* property does not exceed the maximum of 24 characters.

For more information about the syntax, see [Syntax for Service Descriptor](#).

- The key user must have the business role SAP\_BR\_ADMINISTRATOR (Administrator) that contains the business catalog SAP\_CORE\_BC\_COM (Communication Management).

## Context

Create a communication arrangement to enable the exchange of events.

### ⓘ Note

Events can only be published if the topic space matches the namespace of the SAP Event Mesh service.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Communication Arrangements* artifact.
3. Choose *New* to create a new communication arrangement.
4. Enter or select the *Scenario* SAP\_COM\_0092 (communication scenario ID) for *Enterprise Eventing Integration*.
5. Adapt the *Arrangement Name*.
6. To specify additional properties, proceed as follows:
  - You can either use the default values and can ignore the following substeps.
  - To maintain additional properties, choose the *Additional Properties* button and see the following substeps.
    - a. Specify the *Channel* name. The *Channel* name is required in the next steps to configure topic bindings.

### ⓘ Note

Optional. If the *Channel* name is not specified, the *Channel* name equals the *Arrangement Name*. The channel name is used as a key in all enterprise event enablement SAP Fiori applications.

- b. Enter a *Description*.

### ⓘ Note

Optional. If you don't enter a description, this field will be filled with a default description similar to **Channel CHANNEL\_NAME for Enterprise Event Enablement**.

- c. Enter the *Topic Space*.

### ⓘ Note

Optional. The *Topic Space* is extracted from the service key.

- d. Enter the *QoS* (Quality of Service) value. The default QoS value is 1.

### ⓘ Note

Optional.

0	1
At most once delivery.	At least once delivery.
The message is delivered according to the capabilities of the underlying network. No response is sent by the receiver and no retry is performed by the sender. The message arrives at the SAP Event Mesh either once or not at all.	This Quality of Service ensures that the message arrives at the SAP Event Mesh at least once.

- e. Enter the *Reconnect Attempts* value. The default value of the reconnection attempts is 0.

#### ⓘ Note

Optional. The reconnect attempts value is the number of attempts the enterprise event enablement framework tries to reestablish the connection if the connection is lost.

If the value is 0, the framework tries to reconnect infinitely (until the connection is reestablished).

If any value greater than 0 is specified, the framework tries to reestablish the connection as many times as specified. If the connection is not established after reaching the specified reconnect attempts, the communication arrangement and the underlying channel are deactivated. You can reactivate the communication arrangement and channel by selecting the corresponding communication arrangement and choosing *Reactivate*.

- a. Define the *Reconnect wait time(sec)*. The default wait time for reconnection is 10 seconds.

#### ⓘ Note

Optional. The enterprise event enablement framework waits for the mentioned time (in seconds) and then tries to reconnect. The value entered here is the initial wait time. For all subsequent wait times, the enterprise event enablement framework increases the wait time according to the internal logic. If the attempts fail, the framework increases the wait time until the reconnect wait time (1800 seconds) is reached.

7. Choose *Close* to confirm your entries.

8. Select a *Communication User*.

For information about how to create a communication user, see [Creating Technical Communication User \[page 1250\]](#).

9. Enter the *Service Key*.

10. Choose *Create*.

#### ⓘ Note

The connection is checked automatically. If the check fails, the communication arrangement is not created and the end user is informed about the failure reasons.

The edit page is opened.

11. Optional: In the *Outbound Services* section, under *Delivery of Events*, you can unmark the *Active* checkbox to save the communication arrangement in status "inactive". This can be useful, for example, if you want to complete the allowlisting before activating the communication arrangement. To activate the

communication arrangement at a later time, open the edit screen again and set the mark for the *Active* checkbox.

12. Choose **Save**.

After saving the communication arrangement, the channel is activated and the connection is established. The result of the connection test is displayed in the message view on the bottom left.

## Results

You have created a communication arrangement.

**ⓘ Note**

To maintain inbound and outbound bindings, refer to [Configuration of Event Publishing and Event Consumption Scenarios \[page 1274\]](#)

### 4.2.9.11.4.1.2 Maintaining Communication Arrangements

In this topic, a communication arrangement is maintained.

#### Prerequisites

The key user must have the business role SAP\_BR\_ADMINISTRATOR (Administrator) that contains the business catalog SAP\_CORE\_BC\_COM (Communication Management).

#### Context

After creating a communication arrangement, you can still change parameters, either via a service key or by entering parameters manually. You can change the default values for the Quality of Service (QoS), the number of reconnect attempts, the reconnect wait time, and the topic space.

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Communication Arrangements* artifact.
3. Select the communication arrangement you want to maintain.

#### Update by Service Key

4. To change connection details, click the *Update by Service Key* button.
5. Enter the service key with the new connection details.
6. Choose *Update* to save your entries.

#### Update Additional Properties

7.  Note

Updating via the additional properties reactivates the communication arrangement and channel (if deactivated).

To change additional properties, choose *Edit*.

8. To save your changes, choose *Save*.

## Results

You have maintained a communication arrangement.

### 4.2.9.11.4.1.3 Deleting Communication Arrangements

In this topic, a communication arrangement is deleted.

#### Prerequisites

The key user must have the business role `SAP_BR_ADMINISTRATOR` (Administrator) that contains the business catalog `SAP_CORE_BC_COM` (Communication Management).

#### Context

After creating a communication arrangement, you can still delete the communication arrangement.

#### Procedure

1. Log on to the SAP Fiori launchpad in the SAP BTP, ABAP environment system.
2. In the *Communication Management* app, select the *Communication Arrangements* artifact.
3. Select the checkbox of the communication arrangement you want to delete.
4. Choose *Delete*.

## Results

You have deleted a communication arrangement.

### ⓘ Note

When a communication arrangement is deleted, all events belonging to the channel – including those that have not yet been successfully published to the respective service – are deleted from the event queue. Additionally, the configuration for the underlying channel, such as inbound and outbound bindings, is deleted.

## Next Steps

After deleting the communication arrangement, you need to delete the communication system belonging to the communication arrangement manually.

### 4.2.9.11.4.2 Queue Subscriptions

Events published through an ABAP environment instance can be consumed at the SAP Event Mesh. Events published to a queue defined in your SAP Event Mesh service instance can also be consumed in ABAP environment. Queues can be used to buffer events until a consumer can process them. For the right events to arrive at a queue, the queue must be subscribed to the corresponding topics. A consumer can then subscribe to the queue.

## Context

Once a queue has been subscribed to the corresponding topic and you have subscribed to the queue, you can view the list of applications and service instances. The SAP Event Mesh dashboard enables you to create queues and bind incoming message topics to such queues.

### ⓘ Note

The syntax used to compose topics and the payload for ABAP environment business events supports the standardized **CloudEvents** format (<https://cloudevents.io>).

For more information about the syntax, see also [Syntax for Service Descriptor](#).

Ensure that the pattern of the topic, to which the queue is subscribed, consists of the following three parts:

Name	Description
Topic namespace	<ul style="list-style-type: none"><li>Contains three segments: vendor, product information, and a technical identifier.</li><li>The topic namespace originates from the namespace of the service key created for the respective SAP Event Mesh instance.</li></ul>
Abbreviation	Value <code>ce</code> for cloud events
Event topic	Value as displayed in the value help in the previous <i>Outbound Topic Binding</i> configuration step.

You can also use wildcards, such as `<namespace>/*`.

#### ❖ Example

If you want to create a queue-topic subscription for the `sap/s4/beh/businesspartner/v1/ BusinessPartner/Changed/v1` event topic, you need to add the topic namespace and the `ce` abbreviation at the beginning of the topic: `<namespace>/ce/sap/s4/beh/businesspartner/v1/ BusinessPartner/Changed/v1`.

## Related Information

[Manage Queue Subscriptions](#)

[Configuration of Event Publishing and Event Consumption Scenarios \[page 1274\]](#)

[CloudEvents Context Attributes \[page 1251\]](#)

## 4.2.9.11.5 Integration with Advanced Mesh Service Plan for Service Cloud

With the SAP Event Mesh Advanced Plan, you can connect directly to your own SAP AEM instance. You can use SAP Event Mesh Advanced Plan for very large projects, where performance is crucial.

If you have a valid subscription to the SAP Integration Suite, advanced event mesh, you can use the Virtual Message Router (VMR) endpoint of SAP AEM broker obtained from the SAP AEM Console UI. The SAP AEM's validation service then only checks if a valid SAP AEM broker instance is available. If a valid SAP AEM instance is available, a direct connection to this instance is established.

This section describes the integration of the enterprise event enablement with the communication scenarios *SAP Advanced Event Mesh Integration* (SAP\_COM\_0492) and *SAP Advanced Event Mesh Validation Service Integration* (SAP\_COM\_0493).

The communication scenario SAP\_COM\_0492 is the prerequisite for SAP\_COM\_0493. To integrate with the SAP Advanced Event Mesh service, you need to create both the scenarios.

For the integration of an ABAP Platform Cloud tenant, you require a subscription to the SAP Advanced Event Mesh service and to the SAP AEM Validation service.

### 4.2.9.11.5.1 Uploading the Client Certificate

#### Context

To connect your ABAP environment system with the SAP AEM broker, you need to upload the client certificate and the root certificate to the SAP AEM broker.

#### Procedure

1. On the SAP Fiori launchpad in the ABAP environment system, open the app [Maintain Client Certificate](#).
2. To download the **client certificate**, proceed as follows:
  - a. Open the [Client Default](#) from the [Client Certificates](#) list.
  - b. To export the **client certificate**, choose [Export](#).
  - c. On the [Export Certificate](#) dialog box, choose the [X.509 Certificate \(.pem\)](#) format.
  - d. Choose [Export](#).

 Note

Remove any line breaks from the downloaded certificate before you proceed.

3.  Note

You need both the client and the root certificate. The root certificate can be extracted from the client certificate. To access the root certificate, you must download the client certificate again in the crt-format and then extract the root certificate from this file.

To prepare the **root certificate**, proceed as follows:

- a. In the [Maintain Client Certificate](#) app, for the [Client Default](#), choose [Export](#) again.
- b. On the [Export Certificate](#) dialog box, choose the [Base-64-encoded X.509 certificate \(.crt\)](#) format.

 Note

The root certificate must be extracted from the crt-file. The procedure depends on whether you work on a [Windows](#) or on a [Mac](#) computer.

4. To prepare the **root certificate** on a [Windows](#) computer, proceed as follows:

- a. Double-click on the downloaded crt-file, or right click on the file and choose *Open*.
  - b. Open the *Details* tab.
  - c. Choose the *Copy to File...* button, then choose *Next*.
  - d. Select the *Base-64 encoded X.509 (.CER)* format, and choose *Next*.
  - e. Enter a file name and browse for a location, then choose *Next*.
  - f. Choose *Finish*.
5. To prepare the Root Certificate on a *Mac* computer, proceed as follows:
    - a. Open the *Keychain* app.
    - b. Search for SAP.
    - c. Select the correct root certificate.

Make sure to select the correct root certificate.

    - d. Right-click on it and select *Export <name of certificate>*.
    - e. Select the *Privacy Enabled Mail (.pem)* format.
    - f. choose *Save*.
  6. To upload the certificates to the SAP AEM broker, proceed as follows:
    - a. Open the SAP AEM broker UI console.
    - b. On the *Manage* tab, choose the *Certificate Authorities* tile.
    - c. Choose the *Add Client Certificate Authority* button.
    - d. Provide a *Certificate Authority Name*.
    - e. Copy the content of the certificate you downloaded from your ABAP environment system before.
    - f. Paste the content into *Client Certificate Content* field.
    - g. Choose *Save*.
  7. To upload the root certificate, proceed in the same way as for the client certificate.

## 4.2.9.11.5.2 Creating a Communication Arrangement for the Advanced Event Mesh Integration Scenario (SAP\_COM\_0492)

### Context

This communication arrangement is the prerequisite for the AEM Validation Service (SAP\_COM\_0493).

### Procedure

To create a communication arrangement, proceed as follows:

1. Log on to the SAP Fiori launchpad in the ABAP environment system.

2. In the *Communication Management* app, select the *Communication Arrangements* artifact.
3. Choose *New*.
4. Enter or select the *Scenario* SAP\_COM\_0492 (communication scenario ID) for *SAP Integration Suite, Advanced Mesh Integration*.
5. Adapt the *Arrangement Name*.
6. Choose *Create*.
  - a. Specify the *Channel* name. The *Channel* name is required in the next steps to configure topic bindings.
  - b. Enter a *Description*.

**ⓘ Note**

Optional.

- c. Enter the *Topic Space*. The topic space is the identifier for events that originate from the same source.

**→ Tip**

You can enable a sender verification for your *Advanced Event Mesh* instance by configuring the access control per client in the *SAP Integration Suite, advanced event mesh*. The following steps are required in your *Advanced Event Mesh* instance:

1. Choose  *Access Control*  *ACL Profiles* .
2. Select the client profile for which you want to configure the ACL profile.
3. Choose *Publish Topic*.
4. Select *Disallow* as *Publish Default Action*.
5. Choose the  *+ Exception* button.
6. Enter the topic space of your event channel in ABAP environment and a wildcard. For example: topic space/> with SMF or topic space/# with MQTT.

For more information, refer to [Using Client Profiles and Client Usernames](#).

**ⓘ Note**

Mandatory. The topic space has to match regex: [A-Za-z0-9]{1,}([/[A-Za-z0-9\-.]{1,}){2}

- Segments: exactly three (firstSegment/secondSegment/thirdSegment)
- Allowed characters:
  - First segment: [a-zA-Z0-9]
    - Hyphens and dots aren't allowed.
    - Recommended value: Default.
  - Second segment: [a-zA-Z0-9-]
    - Starts with a character or number.
    - Followed by . or -
    - Must contain at least one character or number before or after the period or hyphen.
  - Third segment: [a-zA-Z0-9-]
    - Use the same guidelines given for the second segment.
    - Max Length: 24

- d. Enter the *QoS* (Quality of Service) value. The default QoS value is 1.

**ⓘ Note**

Optional.

**0**

At most once delivery.

**1**

At least once delivery.

The message is delivered according to the capabilities of the underlying network. No response is sent by the receiver and no retry is performed by the sender. The message arrives at the SAP AEM broker either once or not at all.

This Quality of Service ensures that the message arrives at the SAP AEM broker at least once.

- e. Enter the *Number of Publish Connections*. Defines the number of available parallel connections to the SAP AEM broker for this channel. The default value is 1, the maximum value is 10.

**ⓘ Note**

Optional.

- f. Enter the *Reconnect Attempts* value. The default value of the reconnection attempts is 0.

**ⓘ Note**

Optional. The reconnect attempts value is the number of attempts the enterprise event enablement framework tries to reestablish the connection if the connection is lost.

If the value is 0, the framework tries to reconnect infinitely (until the connection is reestablished).

If any value greater than 0 is specified, the framework tries to reestablish the connection as many times as specified. If the connection is not established after reaching the specified reconnect attempts, the communication arrangement and the underlying channel are deactivated. You can reactivate the communication arrangement and channel by selecting the corresponding communication arrangement and choosing *Reactivate*.

- g. Enter the *Reconnect wait time* in seconds.

**ⓘ Note**

Optional. The default time is 10 seconds.

To create a communication system, proceed as follows:

7. In the *Common Data* section, press *New* to create a *Communication System*.
8. In the *New Communication System* dialog box, proceed as follows:
  - a. Enter the *System ID*.
  - b. Enter the *System Name*.
  - c. Choose *Create*.
9. In the *General* tab, in the *Host Name* field, enter the host name of the AMQP connectivity endpoint specified in the cluster manager of the broker instance.

In your *AEM UI Console*, open the *Connect* tab. Find the destination URL under  *Connection Details*  **Secured AMQP Host**  Copy the **host name** only.

### ❖ Example

In this destination URL example:

amqps://mr-connection-demo4xyz.messaging.solace.cloud:1234

The host name is: mr-connection-demo4xyz.messaging.solace.cloud

10. Copy the digits from end part of the URL into the *Port* field.

### ❖ Example

In the above mentioned destination URL example, the port number is 1234.

11. In the *Users for Inbound Communication* tab, press **+**.

This user is used to run the background process that keeps the connection up and running.

12. In the *New Inbound Communication User* dialog box, proceed as follows:

- a. Choose the *Authentication Method User Name and Password*.
- b. Enter the *User Name*.
- c. Enter a *Description*.
- d. Choose the *Propose Password* button.
- e. Choose *Create*.
- f. Choose *OK*.

13. In the *Users for Outbound Communication* tab, press **+**.

14. In the *New Outbound User* dialog box, proceed as follows:

- a. Choose the *Authentication Method SSL Client Certificate*.
- b. Choose the *Client Default* for the *Client Certificate* from the value help.
- c. Choose *Create*.

15. Choose *Save* to save the communication system.

16. Choose *Save* to save the communication arrangement.

Note that the connection to the SAP Advanced Event Mesh only becomes active after creation of SAP\_COM\_0493.

## 4.2.9.11.5.3 Creating a Communication Arrangement for the Advanced Event Mesh Validation Service Integration (SAP\_COM\_0493)

### Context

This communication arrangement is needed to validate the SAP AEM broker destination URL.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Communication Arrangements* artifact.
3. Choose *New* to create a new communication arrangement.
4. Enter or select the *Scenario* SAP\_COM\_0493 (communication scenario ID) for *SAP AEM Validation Service*.
5. Adapt the *Arrangement Name*.
6. Choose the *Additional Properties* button.
  - a. For *Channel*, enter the name of the first channel you created in the previous chapter (SAP\_COM\_0492).
7. Choose *Close* to confirm your entries.
8. Enter the *Service Key*.

You find the service key in your SAP BTP Cockpit Subaccount under [View Credentials](#).

9. Choose *Create*.

To check the connection, proceed as follows:

10. In the *Communication Management* app, go to *Communication Arrangements*.
11. Open the SAP\_COM\_0492 channel you created before.
12. Under *Outbound Services*, choose the *Check Connection* button.

The connection is tested. You get a status message from the Fiori Launchpad whether the connection check was successful.

## Results

### ⓘ Note

To maintain inbound and outbound bindings, refer to [Configuration of Event Publishing and Event Consumption Scenarios \[page 1274\]](#)

### 4.2.9.11.6 Direct Push-Based Integration

With direct push-based integration, you can connect neighboring ABAP Systems (for example SAP S/4HANA Public/Private Cloud, SAP S/4HANA On-Premise or SAP BTP, ABAP environment tenants) with each other. In this scenario, the connected systems exchange data (like business events) by calling each other directly via RFC.

Advantages of this functionality are a simpler setup and a faster connectivity (depending on the RFC technology used). This integration can also be used to try out event-driven architecture in a small scope.

### Note

Direct push-based integration can only be used between ABAP systems. An ABAP system can be an SAP S/4HANA Cloud, SAP S/4HANA On-Premise or SAP BTP, ABAP environment system.

For this one-directional process, you need (at least) two ABAP systems:

- one receiver system
- one (or more) sender system(s)

## 4.2.9.11.6.1 Creating the Receiver Channel SAP\_COM\_0A23

### Context

Outline:

- Create a new communication arrangement.
  - Create a new communication system.
    - Create a new inbound user as daemon user. Copy user and password.
    - Save the communication system.
  - Save the communication arrangement. Copy API-url.

### Procedure

1. Open the SAP Fiori launchpad of the receiver system.
2. In the *Communication Management* section, open *Communication Arrangements*.
3. Choose the *New* button.
4. Select the scenario **SAP\_COM\_0A23**, labeled *Enterprise Event Enablement - Direct Event Inbound Integration*.
5. Enter an arrangement name and choose *Create*.  
The communication arrangement editor is opened.
6. In the *Common Data* section, next to the *Communication System* field, choose *New* to create a new communication system.  
The communication system editor is opened.
7. Provide a *System ID* and a *System Name*.  
You can choose any system ID and name. For example, you can use the same as for the communication arrangement for both ID and name.
8. Choose *Create*.

The new *Communication System* is opened.

9. In the *Technical Data* section, under *General*, mark the *Inbound Only* checkbox.
10. In the section *Users for Inbound Communication*, choose the + symbol on the right to create a new user.
  - a. For the *Authentication Method*, select *User Name and Password*.
  - b. Leave the *User Name/Client ID* field blank and choose *New User*.

The *Create Communication User* editor is opened.

- c. Enter a *User Name* and a *Description*.
- d. Enter a *Password* or choose *Propose Password* and save it for later.
- e. Choose *Create*.
- f. Choose *OK* in the *New Inbound Communication User* popup window.

A new inbound user is created and added to the communication system.

11. Choose *Save* in the *Communication System* creation window.
12. In the *Communication Arrangements* window, in the *Additional Properties* tab, provide a *Channel* and a *Description*.

You can choose any channel name. For example, you can use the same as for the communication arrangement for both channel and description. This channel name is used later during the channel configuration and monitoring.

13. From the *Common Data* section, copy the provided *API-URL* without `https://`. You will need this info later for the host name of the sender channel.
14. Choose *Save*.

## 4.2.9.11.6.2 Creating the Sender Channel SAP\_COM\_0A22

### Context

Outline:

- Create a new communication arrangement
  - Create a new communication system.
    - Create a new outbound user: choose the daemon user created before.
    - Additionally, create a new inbound user as destination.
  - Save the communication system.
- Save the communication arrangement.

### Procedure

1. Open the SAP Fiori launchpad of the sender system.

2. In the *Communication Management* section, open *Communication Arrangements*.
3. Choose the *New* button.
4. Select the scenario SAP\_COM\_0A22, labeled *Enterprise Event Enablement - Direct Event Outbound Integration*.
5. Enter an arrangement name and choose *Create*.

The communication arrangement editor is opened.
6. In the *Common Data* section, next to the *Communication System* field, choose *New* to create a new communication system.

The communication system editor is opened.
7. In the *General* section, paste the API-url you created in the communication arrangement of receiver channel into the *Host Name* field.

Paste the url without `https://`.

For more information about where to find the API-url for the host name, see [Creating the Receiver Channel SAP\\_COM\\_0A23 \[page 1269\]](#).
8. In the section *Users for Outbound Communication*, choose the *+* symbol on the right to create a new user.
  - a. For the *Authentication Method*, select *User Name and Password*.
  - b. In the *User Name/Client ID* field, enter the inbound user you created in the receiver channel.
  - c. Enter the *Password* you defined for the inbound user of the receiver channel.
  - d. Choose *Create*.

A new outbound user is added to the communication system.
9. In the section *Users for Inbound Communication*, choose the *+* symbol on the right to create a new user.
  - a. For the *Authentication Method*, select *User Name and Password*.
  - b. Leave the *User Name/Client ID* field blank and choose *New User*.

The *Create Communication User* editor is opened.

  - c. Enter a *User Name* and a *Description*.
  - d. Enter a *Password* or choose *Propose Password*.
  - e. Choose *Create*.
  - f. Choose *OK* in the *New Inbound Communication User* popup window.

A new inbound user is created and added to the communication system.
10. In the *Communication Arrangements* window, in the *Additional Properties* tab, provide a *Channel* and a *Description*.

You can choose any channel name. For example, you can use the same as for the communication arrangement for both channel and description.
11. In the *Communication Partner ID* field, enter the name of the SAP\_COM\_0A23 receiver channel you created before.
12. Choose *Save*.

The communication arrangement is activated.

## 4.2.9.11.6.3 Configuring the Receiver

### Context

Create a new communication arrangement representing a consumer on the receiver side.

Outline:

- Create a new communication arrangement.
  - Create a new communication system.
    - Mark checkbox *Inbound Only*
    - Set *Event Exchange Infrastructure* slider to *On*.
      - Add the SAP\_COM\_OA23 receiver event channel.
    - Create a new inbound user.
    - Save the communication system.
  - Save the communication arrangement.
  - Open the receiver channel to see all inbound topics listed.

### Procedure

1. Open the SAP Fiori launchpad of the receiver system.
2. In the *Communication Management* section, open *Communication Arrangements*.
3. Choose the *New* button.
4. Select a suitable consumption scenario.
5. Enter an arrangement name and choose *Create*.

The communication arrangement editor is opened.
6. In the *Common Data* section, next to the *Communication System* field, choose *New* to create a new communication system.

The communication system editor is opened.
7. Provide a *System ID* and a *System Name*.

You can choose any system ID and name. For example, you can use the same as for the communication arrangement for both ID and name.
8. Choose *Create*.

The new *Communication System* is opened.
9. In the *Technical Data* section, under *General*, mark the *Inbound Only* checkbox.
10. In the *OAuth 2.0 Settings* section, set the *Event Exchange Infrastructure* slider to *On*.
11. In the *Event Exchange Infrastructure* section, choose the *+* symbol on the top right to add an event channel.
12. Select the SAP\_COM\_OA23 receiver event channel you created earlier. (See: [Creating the Receiver Channel SAP\\_COM\\_OA23 \[page 1269\]](#))

13. In the section *Users for Inbound Communication*, choose the **+** symbol on the right to create a new user. You can select an existing user from the value help or create a new one. To create a new user, proceed as follows:

- a. For the *Authentication Method*, select *User Name and Password*.
- b. Leave the *User Name/Client ID* field blank and choose *New User*.  
The *Create Communication User* editor is opened.
- c. Enter a *User Name* and a *Description*.
- d. Enter a *Password* or choose *Propose Password* and save it for later.
- e. Choose *Create*.
- f. Choose *OK* in the *New Inbound Communication User* popup window.

A new inbound user is created and added to the communication system.

14. Choose *Save* in the *Communication System* creation window.

15. Choose *Save* in the *Communication Arrangement* window.

## Results

In the receiver channel, all defined inbound topics bindings are now displayed. You can now proceed to the following topic to syncronize the sender with the receiver channel.

### 4.2.9.11.6.4 Synchronizing Sender and Receiver Channels

## Context

Outline:

- Open sender channel, outbound topic bindings.
  - Choose button *Overwrite with remote topic bindings*. The available outbound topics are called from the receiver and set up automatically. Sender and receiver are now synchronized.
- Test your connection: Go to *Event Monitor*. Mark your sender channel. Choose *Produce Test Event*. Refresh the page.

## Procedure

1. Open the SAP Fiori launchpad of the sender system.
2. Open the app *Enterprise Event Enablement - Configure Channel Binding*.
3. Open your sender channel.

The *Outbound Topic Bindings* list is empty if it hasn't been synchronized before.

4. Choose the *Overwrite with remote topic bindings* button on the top right.

The available outbound topics are called from the receiver channel and set up automatically in the sender channel.

#### Note

If the corresponding event topic cannot be assigned, for example because the event topic doesn't exist in the system, a corresponding message is displayed.

If the receiver receives events that are not listed in the inbound binding allowlist, these events are stored in the receiver channel in status "failed".

5. **[Optional]** To test your connection, go to the *Event Monitor*.

- a. Mark your sender channel and choose *Produce Test Event*.
- b. Refresh the page.

The test event is counted in the column *In Process Events* column.

## 4.2.9.11.7 Configuration of Event Publishing and Event Consumption Scenarios

You can define which event types shall be published or consumed using a connection defined through the communication arrangement. Each event type is assigned to one topic. Topics form a logical tree to organize events, such as a folder hierarchy in a file system. Thus, the topics appear as strings that consist of multiple segments and are separated by one defined delimiter, similar to file paths.

#### Note

Only events, which are bound to a connection, can be exchanged between an ABAP environment system and the SAP Business Technology Platform.

The following steps are required to configure event publishing and event consumption scenarios:

[Event Publishing \[page 1274\]](#)

[Event Consumption \[page 1279\]](#)

[Event Metadata \[page 1284\]](#)

## 4.2.9.11.7.1 Event Publishing

The following chapters describe how to publish events to a consumer using outbound event topics. You can also use filters to determine the type of events that are published.

### Note

Event publishing is only relevant for the integration with SAP Event Mesh and Advanced Event Mesh. It is not relevant for the integration with SAP Cloud Application Event Hub.

## 4.2.9.11.7.1.1 Maintain Outbound Event Topics

### Prerequisites

- You have established a connection through the configuration of the communication arrangement in the previous steps. For more information, refer to [Communication Arrangements \[page 1256\]](#).
- The *Enterprise Event Enablement – Configure Channel Bindings* app is visible.  
If the *Enterprise Event Enablement – Configure Channel Bindings* app is not visible, add the *Enterprise Event Enablement* business catalog to the `SAP_BR_ADMINISTRATOR` role using the *Maintain Business Roles* app. The change in your role becomes effective after you have logged out and logged in again.

### Context

You can configure outbound event topic bindings for publishing events.

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Enterprise Event Enablement – Configure Channel Binding* app.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)).
4. In the *Outbound Topic Bindings* tab, you can do the following:

Option	Description
<b>Create a new topic.</b>	<ol style="list-style-type: none"><li>1. Choose <i>Create</i>.</li><li>2. Enter a topic name.</li><li>3. Choose <i>Create</i> to save your entries.</li></ol> <p>The new topic will be listed in the <i>Topic</i> table.</p>
<b>Delete an existing topic.</b>	<ol style="list-style-type: none"><li>1. Select the checkbox of the topic you want to delete.</li><li>2. Choose <i>Delete</i>.</li></ol>

Option	Description
Define table parameters.	Click the gear icon to define the table parameters.

## Next Steps

Events reside in an internal queue of the enterprise event enablement. Once the processing of an event is finalized, it is deleted from the queue. The processing of an event is finalized if:

Status: Ok	The event has been successfully published according to the chosen quality of service.
Status: Failed	The enterprise event enablement failed to publish the event after 5 attempts.

## Related Information

[Queue Subscriptions \[page 1261\]](#)

## 4.2.9.11.7.1.2 Maintain Filters for Outbound Event Topics

You can filter the events that should be sent via a channel by defining filters per channel and topic.

### Prerequisites

- You have created an outbound event topic as described here: [Maintain Outbound Event Topics \[page 1275\]](#)
- **ⓘ Note**

Event filters for event topics containing wildcards are not supported.

### Context

You can define filters for context and extension context attributes defined for the corresponding event type. In the [RESTful Application Programming](#) framework (RAP), you can define a custom context attribute for your RAP business object with the CDS annotation `@event.context.attribute`.

For more information on the CDS annotation, see: [Event Annotations](#)

The custom defined context attribute is then available as a property in the event filter creation wizard in the [Enterprise Event Enablement – Configure Channel Binding](#) app.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the [Communication Management](#) app, select the [Enterprise Event Enablement – Configure Channel Binding](#) app.
3. Choose a channel.

In the [Outbound Topics](#) table, the column [Has Event Filters](#) indicates whether filters are defined for the outbound topic.

4. Open an [Outbound Topic](#).  
Choose the [Event Filter](#) tab to display existing filters for the outbound topic.
5. To add a filter to the outbound topic, choose [Create Event Filter](#).
6. On the [Create Event Filter](#) dialog box, proceed as follows:
  - a. Select a [Property](#) from the dropdown menu. The properties available in the dropdown menu are defined by the extension attributes.
  - b. Choose an [Option](#) from the dropdown menu to define the filter behavior.
  - c. Define a [Low Value](#). You can use the value help, if available for the respective property.
  - d. If applicable, define a [High Value](#).
  - e. Choose [Create Event Filter](#).

The new filter is added to the outbound topic. In accordance with existing filters, the new filter is added with an [OR](#) logic for inclusive options such as [EQUALS](#). New filters are added with an [AND](#) logic for exclusive options such as [NOT EQUALS](#). Different Properties are always [AND](#) connected.

To delete a filter from an outbound event topic, select one or more filters and choose the [Delete](#) button.

### ⓘ Note

For more information on the usage of event filters, refer to [3343266](#).

## 4.2.9.11.7.1.3 Using Dynamic Topics in Advanced Event Mesh

You can use dynamic topics for routing and filtering outbound events in the [Advanced Event Mesh](#) (AEM).

### Prerequisites

#### ⓘ Note

Dynamic topics are available in the [Advanced Event Mesh](#) (AEM) only.

For more information about the use of events in the context of AEM, see [Understanding Topics](#).

- You have created an outbound event topic as described here: [Maintain Outbound Event Topics \[page 1275\]](#)
- You have implemented `event.context.attribute` annotations to define dynamic topic segments for the given event type, as described in [Event Annotations](#).
- In addition, you have implemented `event.context.position` annotations to specify the position of the corresponding dynamic topic segment relative to the other dynamic segments.

## Context

Dynamic topics make use of the `context` attribute annotations to extend the corresponding event topic with further dynamic segments, derived from the corresponding annotated payload field. Dynamic topics enable detailed event filtering and routing, ensuring that events are delivered only to the relevant components, improving efficiency and reducing unnecessary processing.

The [Enterprise Event Enablement](#) framework (EEE) enables the usage of dynamic topics for the [Advanced Event Mesh](#) integration (AEM). A possible example for such a dynamic topic would be as follows:

```
sap/custom/MySalesOrder/v1/{companyCode}/{SalesOrderID}
```

```
----- properties -----
      to: sap.custom.MySalesOrder.Created.v1.de.10002
      content-type: application/cloudevents+json; charset=utf-8

----- application-properties -----

----- application-data -----
{
  "specversion": "1.0",
  "type": "sap.custom.MySalesOrder.Created.v1"
  "xsapcompanycode": "de"
  "xsapsalesorder": 1002,
  "data": { ... }
}
```

### ⓘ Note

Note that the dynamic topic segment values are also part of the CloudEvent header as defined in the `event.context.attribute` annotation.

## Procedure

To enable dynamic topics, proceed as follows:

1. Log on to the SAP Fiori launchpad.
2. In the [Communication Management](#) app, open the [Enterprise Event Enablement – Configure Channel Binding](#) app.
3. Select the corresponding AEM event channel.

4. Choose the *Enable Dynamic Topics* button on the top right.

 Note

When you enable dynamic topics, topics containing wildcards will be resolved, as dynamic segments are specific to a single event topic.

In the *Outbound Topic Bindings* section, the outbound topic bindings are now displayed with the derived dynamic attributes in curly brackets.

 Example

```
sap/custom/MySalesOrder/v1/{companyCode}/{SalesOrderID}
```

The *Use Dynamic Topics* indicator on the top left changes to Yes.

5. To disable dynamic topics, choose the *Disable Dynamic Topics* button on the top right.

## 4.2.9.11.7.2 Event Consumption

Event consumption can be configured manually for some event types (for example subscription billing). These configuration options are described in this section. For all other event types, follow the steps described in [Creating an Event Consumption Model](#).

You can configure the consumption of events. To do so, proceed as follows:

[Configuring Inbound Event Topic Bindings \[page 1279\]](#)

[Configuring Subscriptions \[page 1281\]](#)

[Simulating Consumption \[page 1283\]](#)

### 4.2.9.11.7.2.1 Configuring Inbound Event Topic Bindings

#### Prerequisites

- You have established a connection through the configuration of the communication arrangement in the previous steps. For more information, refer to [Communication Arrangements \[page 1256\]](#).
- The *Enterprise Event Enablement – Configure Channel Bindings* app is visible.  
If the *Enterprise Event Enablement – Configure Channel Bindings* app is not visible, add the *Enterprise Event Enablement* business catalog to the SAP\_BR\_ADMINISTRATOR role using the *Maintain Business Roles* app. The change in your role becomes effective after you have logged out and logged in again.

## Context

You can configure event consumers delivered by SAP manually.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Enterprise Event Enablement – Configure Channel Binding* app.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)).
4. In the *Inbound Topic Bindings* tab, you can do the following:

Option	Description
<b>Create a new topic.</b>	<ol style="list-style-type: none"><li>1. Choose <i>Create</i>.</li><li>2. Enter a topic name.</li><li>3. Choose <i>Create</i> to save your entries.</li></ol> <p>The new topic is listed in the <i>Inbound Topics</i> table.</p>
<b>Delete an existing topic created by an end user.</b>	<ol style="list-style-type: none"><li>1. Select the checkbox of the topic you want to delete.</li><li>2. Choose <i>Delete</i>.</li></ol>
<b>Define table parameters.</b>	Click the gear icon to define the table parameters.

## Next Steps

The *Inbound Topics* table contains all inbound event topic bindings that exist for the selected channel and communication arrangement. The fields can have the following values:

<b>Status: Ok</b>	
<b>Status: Incomplete</b>	No topic consumer exists for current topic binding.
<b>Status: Invalid</b>	The current topic binding is invalid for at least one topic consumer. This means either the topic filter doesn't exist or the corresponding destination is incorrect.
<b>Topic</b>	Is the topic presentation.
<b>Maintained by: End User</b>	The event inbound topic binding was created manually by a user.
<b>Maintained by: Communication Management</b>	The event inbound topic binding was created automatically during the configuration of a generated event consumption model.

### ⓘ Note

In case invalid bindings maintained by an end user exist, you can delete this binding by:

- choosing *Delete invalid bindings* on top of the page or
- selecting the respective topic and choosing *Delete*.

#### ⓘ Note

To enable the consumption of events, the AMQP protocol replaces the MQTT protocol. New communication arrangements use the AMQP protocol by default. Existing communication arrangements can be updated via the service key. To do so, refer to [Maintaining Communication Arrangements \[page 1259\]](#)

## 4.2.9.11.7.2.2 Configuring Subscriptions

### Prerequisites

- You have established a connection through the configuration of the communication arrangement in the previous steps. For more information, refer to [Communication Arrangements \[page 1256\]](#).
- The *Enterprise Event Enablement – Configure Channel Bindings* app is visible.  
If the *Enterprise Event Enablement – Configure Channel Bindings* app is not visible, add the *Enterprise Event Enablement* business catalog to the SAP\_BR\_ADMINISTRATOR role using the *Maintain Business Roles* app. The change in your role becomes effective after you have logged out and logged in again.

#### ⓘ Note

Subscriptions are only available for SAP Event Mesh and Advanced Event Mesh.

### Context

Subscriptions, which are based on a queue, provide all incoming events. These incoming events are forwarded to the corresponding consumer.

#### ⓘ Note

Events are only forwarded to an event consumer when an inbound event topic binding exists for the related event topic.

## Procedure

1. Log on to the SAP Fiori launchpad in the SAP S/4HANA Cloud system.
2. In the *Communication Management* app, select the *Enterprise Event Enablement – Configure Channel Binding* app.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)).
4. In the *Subscriptions* tab, you can do the following:

Option	Description
<b>Create a new subscription.</b>	<ol style="list-style-type: none"><li>1. Choose <i>Create</i>.</li><li>2. Enter a valid subscription address that corresponds to the name of the queue created in the respective event exchange infrastructure.</li><li>3. Choose <i>Create</i> to save your entries.</li></ol> <p>The new subscription is listed in the <i>Subscriptions</i> table.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p><b> ⓘ Note</b></p><p>When working with SAP Event Mesh, you can create a maximum of 10 subscriptions.</p></div>
<b>Delete an existing subscription.</b>	<ol style="list-style-type: none"><li>1. Select the checkbox of the subscription you want to delete.</li><li>2. Choose <i>Delete</i>.</li></ol>
<b>Define table parameters.</b>	Click the gear icon to define the table parameters.

## Next Steps

Subscriptions must have a valid subscription address provided by the respective event exchange infrastructure. Both subscriptions and inbound bindings are required to consume events. For more information, refer to [Configuring Inbound Event Topic Bindings \[page 1279\]](#).

The *Subscriptions* table contains all subscriptions that exist for the selected channel. The different fields can have the following values:

<b>Status: New</b>	Subscription was newly created and has not been acknowledged by the respective event exchange infrastructure service instance yet.
<b>Status: Acknowledged</b>	Subscription is active and acknowledged by the respective event exchange infrastructure service instance.
<b>Status: Rejected</b>	The channel has been deactivated and the subscription is rejected by the respective event exchange infrastructure service instance.
<b>Status: Inactive</b>	Subscription wasn't acknowledged by the respective event exchange infrastructure service instance.
<b>Subscription Address</b>	Depicts the current address.

Type Kind	Queue-Based Subscription
-----------	--------------------------

### 4.2.9.11.7.2.3 Simulating Consumption

#### Prerequisites

- The [Enterprise Event Enablement – Configure Channel Bindings](#) app is visible.  
If the [Enterprise Event Enablement – Configure Channel Bindings](#) app is not visible, add the [Enterprise Event Enablement](#) business catalog to the SAP\_BR\_ADMINISTRATOR role using the [Maintain Business Roles](#) app.  
The change in your role becomes effective after you have logged out and logged in again.
- You have an inbound topic binding with status [Ok](#) in your [Inbound Topics](#) table.

#### Context

To allow you to test your consumer implementation, you can use the [Simulate Consumption](#) action. Then a dummy event of a given [Topic Consumer](#) will be sent to its consumer implementation.

#### Procedure

1. Log on to the SAP Fiori launchpad in the SAP S/4HANA Cloud system.
2. In the [Communication Management](#) app, select the [Enterprise Event Enablement – Configure Channel Binding](#) app.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)).
4. Select an [Inbound Topic](#).  
If you have no [Inbound Topic](#) with [Status: Ok](#), refer to [Configuring Inbound Event Topic Bindings \[page 1279\]](#)
5. Select the radio button of the [Topic Consumer](#) you want to simulate consumption with.
6. Choose [Simulate Consumption](#).

#### Results

In case of a wildcard in the topic, the wildcard is resolved and events for all resolved topics are created. The dummy events are sent to the corresponding consumer implementation. Then the simulated event is retrieved from the internal event queue and the status is checked. A short message is raised, informing the user on the success or failure of the simulation.

## 4.2.9.11.7.3 Event Metadata

### Prerequisites

- You have established a connection through the configuration of the communication arrangement in the previous steps. For more information, refer to [Communication Arrangements \[page 1256\]](#).
- The *Enterprise Event Enablement – Configure Channel Bindings* app is visible.  
If the *Enterprise Event Enablement – Configure Channel Bindings* app is not visible, add the *Enterprise Event Enablement* business catalog to the SAP\_BR\_ADMINISTRATOR role using the *Maintain Business Roles* app. The change in your role becomes effective after you have logged out and logged in again.

### Context

The enterprise event enablement is applying the CloudEvents specification (<https://cloudevents.io>) for the runtime representation of an event. CloudEvents is a specification for describing event data in common formats to ensure interoperability across services, platforms, and systems.

You can find all released SAP events on the SAP Business Accelerator Hub (<https://api.sap.com>). The SAP Business Accelerator Hub is the central catalog of SAP and selected partner APIs to search, discover, test, and consume these APIs to build extensions or integrations using the SAP Business Technology Platform.

The enterprise event enablement applies the AsyncAPI specification to describe a catalog of events.

#### ⓘ Note

You can find the corresponding AsyncAPI event catalogs for released events in the *Event Resources* section of the individual released events on the SAP Business Accelerator Hub. This specification is required for example to create an event consumption model.

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Enterprise Event Enablement – Configure Channel Binding* app.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)).
4. Select *Event Metadata*.

## Next Steps

In the [Event Metadata](#) tab, you can view the metadata for the event types selected on the [Outbound Topic Bindings](#) tab for event publishing. This metadata contains the definition of the structure of all event types, which you have selected for publishing from the ABAP environment system. The metadata is represented as an AsyncAPI catalog of events and can contain released SAP events as well as customer events. You can also use the AsyncAPI representation for creating your event consumption model.

## Related Information

[CloudEvents Context Attributes \[page 1251\]](#)

[Creating an Event Consumption Model](#)

### 4.2.9.11.8 Read Access Logging (RAL)

With [Read Access Logging](#) (RAL) for events, you can monitor and log read access to sensitive data because of diverse data privacy regulations. The categorization of sensitive data can vary: it can depend on your country's code of law or on your external or internal company policies. RAL thus helps you to find out who had access to sensitive data, in which way and at what time.

RAL is offered for different channels. A channel is a way for data to leave the system, for example dynpros of Web services. The enterprise event enablement is another channel that is supported in the central RAL monitor.

For more information on RAL in the context of SAP NetWeaver, see [Read Access Logging](#).

#### ⓘ Note

Using RAL requires system resources. This can have a strong impact on your system performance, depending on the number and type of logs.

Events can be logged for two scenarios:

- Read access of events during support (monitoring)  
This is the case when the event monitor was opened by SAP support to check the payload of sent or consumed events.
- Read access of events during sending (runtime)  
This is the case when an outbound binding exists and an event is to be sent to the respective event exchange infrastructure by the enterprise event enablement. For more information, refer to [Configuration of Event Publishing and Event Consumption Scenarios \[page 1274\]](#).

You can define which properties of an event are to be logged to avoid that all events are logged. Consider this carefully as it may affect the system performance.

The following types can be logged once a corresponding configuration exists:

Channel Fields	Condition Fields	Access Context
	Cloud Event Context Fields	<ul style="list-style-type: none"><li>• Source</li><li>• Type</li><li>• Id</li><li>• Time</li></ul>
	Payload Types	Payload types of a selected producer or consumer. A producer or consumer is a set of event types belonging to a business entity.
System Fields	User Name	
	Screen Title	
	Transaction code	

#### 4.2.9.11.8.1 Searching for Existing Configurations

##### Prerequisites

The key user must have the business role SAP\_BR\_ADMINISTRATOR (Administrator) that contains the business catalog SAP\_CORE\_BC\_COM (Communication Management).

##### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Read Access Logging Configuration* app, select the *Administration* tab.
3. Choose *Configuration*.
4. Choose *Enterprise Event Enablement* as *Channel*.
5. Specify your *Search Criteria*:
  - Registration ID
  - Registration Version
  - Repository ID
  - Event Scenario
  - Description
  - State
6. Click *Search*.

## Results

A list of existing configurations is displayed.

### 4.2.9.11.8.2 Creating Configurations

#### Prerequisites

The key user must have the business role SAP\_BR\_ADMINISTRATOR (Administrator) that contains the business catalog SAP\_CORE\_BC\_COM (Communication Management).

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Read Access Logging Configuration* app, select the *Administration* tab.
3. Choose *Configuration*.
4. Choose *Enterprise Event Enablement* as *Channel*.
5. Click *Create* at the bottom of the page.
6. On the *Create Configuration* screen, you can search for existing event consumers or event producers to which you can assign a configuration. To do so, specify your search criteria:
  - Registration ID
  - Registration Version
  - Repository ID
  - Event Scenario
  - Busi Object Name

- a. Click *Search*.

A list of existing producers (P) and consumers (C), matching your search criteria, is displayed.

**① Note**

If you do not specify your search criteria and click *Search*, all existing producers and consumers are displayed.

- b. Choose one producer or consumer.
- c. Click *Create*.

For the chosen consumer or producer, the following sections are displayed:

- Attributes*
- *Application/Software Component* that is assigned to a consumer or producer
  - Description for the *Application/Software Component*
- 

- Administrative Information*
- *Technical Data*
  - *Created*
  - *Changed*
- 

*Log Groups*

Log groups are groups of fields that are displayed in one log entry in the read access log.

To add a log group, do the following:

- Click the button with the tooltip *Create Log Group*.
- In the *Create Log Group* dialog box, do the following:
  - Define the *Purpose*.
  - Define the *Description*.
- Click *Create*.

 **Note**

Keep in mind to drag and drop *Channel Fields* and/or *System Fields* from the *Field list* to the *Fields* section of the log group.

 **Tip**

You can create log groups without adding conditions. If you want to add conditions, uncheck the *Without Condition* flag.

*Conditions*

To add a condition, do the following:

- Click the button with the tooltip *Create Condition*.
- In the *Create Condition* dialog box, do the following:
  - Define the *Condition*.
  - Define the *Description*.
- Click *Create*.

*Field list*

---

- d. Click *Save as Active* to activate your configuration.

## Related Information

[Defining Conditions for Configurations \[page 1289\]](#)

## 4.2.9.11.8.3 Defining Conditions for Configurations

### Prerequisites

You've created a configuration in the *Read Access Logging Configuration* app. Refer to [Creating Configurations \[page 1287\]](#).

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Read Access Logging Configuration* app, select the *Administration* tab.
3. Choose *Configuration*.
4. Choose *Enterprise Event Enablement* as *Channel*.
5. Search for your configuration in the *Search Criteria* section. Refer to [Searching for Existing Configurations \[page 1286\]](#).
6. Click the *Edit Configuration* icon of the configuration you want to define a condition for.
7. In the *Conditions* section, click the button with the tooltip *Create Condition*.
  - a. Define the *Condition*.
  - b. Define the *Description*.
  - c. Click *Create*.

### Next Steps

Adding conditions is recommended to specify configurations (refer to [Adding Expressions to Conditions \[page 1289\]](#)). This can influence the system performance depending on the logging scenario you have chosen. If you have chosen read access of events during support (monitoring), the system performance is improved. If you have chosen read access of events during sending (runtime), the system performance can become worse.

## 4.2.9.11.8.4 Adding Expressions to Conditions

### Prerequisites

You've created a condition for a configuration in the *Read Access Logging Configuration* app. Refer to [Defining Conditions for Configurations \[page 1289\]](#).

## Context

Once you've created a condition and you want to log events only in a certain context, like at runtime, add an expression to your condition.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
  2. In the *Read Access Logging Configuration* app, select the *Administration* tab.
  3. Choose *Configuration*.
  4. Choose *Enterprise Event Enablement* as *Channel*.
  5. Search for your configuration in the *Search Criteria* section. Refer to [Searching for Existing Configurations \[page 1286\]](#).
  6. Click the *Edit Configuration* icon of the configuration you want to define a condition for.
  7. In the *Conditions* section, select the condition you want to add expressions for.
  8. Click *Create* in the *Expressions* section of the condition.
    - a. Define a name for the *Expression*.
    - b. Choose *Create*.
- The *Attributes* and *Select options* are displayed in the *Details of Expression* section.
9. Drag and drop the *Access Context* condition field from the *Field List* *Channel Fields* *Condition Fields* to the *Select options* section under *Details of Expression*.
  10. Click *Save as Active* to activate your configuration including assigned conditions.

## Related Information

[Defining Configurations in Read Access Logging](#)

## 4.2.9.11.8.5 Monitoring Logged Events

### Prerequisites

The user must have the business role `BR_DATA_PRIVACY_SPECIALIST` (Data Privacy Specialist). If the user doesn't have the business role `BR_DATA_PRIVACY_SPECIALIST`, add the business role in the *Maintain Business Roles* app.

Using the business role BR\_DATA\_PRIVACY\_SPECIALIST, you can block, unblock, and display blocked personal data. Refer to [Access Control and Data Protection](#).

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
  2. Select the *Read Access Logging: Monitor* app.
  3. Choose *Raw Database as Source* in the *Data Source* section.
  4. Specify your search criteria:
    - Channel
    - Date/Time
    - User Name
    - Application Comp.
    - Software Component
  5. Specify the *Maximum Number of Hits*.
  6. Choose *Search*.
- A list of logs according to the search criteria is displayed.

## 4.2.9.11.9 Message Monitoring

The /IWXBE\_TECHNICAL interface allows you to receive messages in case a technical error occurs. You can monitor the messages using the *Message Monitoring for Integration Experts* app, provided by the SAP Application Interface Framework (SAP AIF).

To display messages in the *Message Monitoring for Integration Experts* app, proceed as follows:

- [Assigning Recipients to Users \[page 1291\]](#)
- [Assigning users to the SAP Application Interface Framework \[page 1293\]](#)
- [Using Message Monitoring \[page 1294\]](#)

### 4.2.9.11.9.1 Assigning Recipients to Users

#### Prerequisites

The user must have the business role BR\_CONF\_EXPERT\_BUS\_NET\_INT (Configuration Expert – Business Network Integration). If the user doesn't have the business role BR\_CONF\_EXPERT\_BUS\_NET\_INT, add the business role in the *Maintain Business Roles* app.

## Context

Before you can display messages in the *Message Monitoring for Integration Experts* app, you must assign a user to a specific recipient.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. Open the *Assign Recipients to Users* app.  
A list containing all users that are already assigned to recipients is displayed.
3. Click the **+** in the bottom row of the left-hand side of the screen.

### ⓘ Note

If you want to add additional recipients to a user who is already assigned to a recipient, use the *Search Users with Assigned Recipients* search field.

4. On the *Add User* dialog box, enter a user name or a user ID.
5. Choose *Add*.
  - a. In the *Assign Recipients* dialog box, enter the following:
    - *Namespace*: /IWXBE
    - *Recipient Name*: RECIPIENT\_CHANNEL
    - *Message Type*: Technical Error
  - b. Choose *Assign* to save the user assignment.  
The assigned recipient is now displayed in the *Recipients for User* list.

## Next Steps

If you want to assign additional recipients to the user, click the *Assign* button on the right hand side of the *Recipients for User* list and repeat the steps.

## 4.2.9.11.9.2 Assigning users to the SAP Application Interface Framework

### Prerequisites

The user must have the business role BR\_CONF\_EXPERT\_BUS\_NET\_INT (Configuration Expert – Business Network Integration). If the user doesn't have the business role BR\_CONF\_EXPERT\_BUS\_NET\_INT, add the business role in the [Maintain Business Roles](#) app.

### Context

In order to receive alerts for the consumption with the given event consumption model, you can assign your user to the SAP Application Interface Framework interface by a corresponding recipient.

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. Open the [Assign Recipients to Users](#) app.  
A list containing all users that are already assigned to recipients is displayed.
3. Click the **+** in the bottom row of the left-hand side of the screen.

#### ⓘ Note

If you want to add additional recipients to a user who is already assigned to a recipient, use the [Search Users with Assigned Recipients](#) search field.

4. On the [Add User](#) dialog box, enter a user name or a user ID.
5. Choose **Add**.
  - a. In the [Assign Recipients](#) dialog box, enter the following:
    - **Namespace:** /IWXBE
    - **Recipient Name:** Choose the generated recipient name.  
The generated recipient name is a <generic\_hash> with the description <event\_consumption\_model\_id> <event\_consumption\_model\_version> ##GENERATED.
    - **Message Type:** Choose a [Message Type](#).

#### → Tip

There is also a general monitoring case with the [Recipient Name RECIPIENT\\_CHANNEL](#). Refer to [Assigning Recipients to Users \[page 1291\]](#).

- b. Choose *Assign* to save the user assignment.

The assigned recipient is now displayed in the *Recipients for User* list.

## Next Steps

If you want to assign additional recipients to the user, click the *Assign* button on the right hand side of the *Recipients for User* list and repeat the steps.

### 4.2.9.11.9.3 Using Message Monitoring

#### Context

Before you can display messages in the *Message Monitoring for Integration Experts* app, you must assign a user to a specific recipient. Refer to [Assigning Recipients to Users \[page 1291\]](#).

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. Open the *Message Monitoring for Integration Experts* app.
  - a. Enter the following parameters:
    - *Time*: Define the period of time to be considered.
    - *From*: Specify the period of time.
    - *To*: Specify the period of time.
    - *Interface*: Enter /IWXBETECHNICAL or select /IWXBETECHNICAL from the combo box.
    - *Additional Interface Options*: Specify the interface that should be displayed.
  - b. Choose *Go*.

The *Message Status Overview* screen displays your interfaces and all error messages with the following states:

- All
- Error
- Warning
- Success
- In Process
- Canceled

## Next Steps

You can click the number of received messages of a specific status to display the corresponding list of messages. Select a message from the *Log Details* section to display the log message details on the right-hand side of the screen. If there is a correlation between the error message and the related communication arrangement, the *Display Communication Arrangement* link is displayed in the *Functions* section. Click the link to open the *Communication Arrangements* app and to display the related communication arrangement.

### Note

You need authorization to open the *Communication Arrangements* app. For more information, refer to [Creating Technical Communication User \[page 1250\]](#).

## Related Information

[Message Monitoring for Integration Experts](#)

### 4.2.9.11.10 Monitoring Events

## Prerequisites

- You have established a connection through the configuration of the communication arrangement in the previous steps. For more information, refer to [Communication Arrangements \[page 1256\]](#).
- The *Enterprise Event Enablement – Event Monitor* app is visible.  
If the *Enterprise Event Enablement – Event Monitor* app is not visible, add the *Enterprise Event Enablement* business catalog to the SAP\_BR\_ADMINISTRATOR role using the *Maintain Business Roles* app. The change in your role becomes effective after you have logged out and logged in again.

## Context

Using the *Enterprise Event Enablement – Event Monitor* app, you can monitor events by status and access the payload.

## Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment system.
2. In the *Communication Management* app, select the *Enterprise Event Enablement – Event Monitor* app. You will see an overview of all configured channels. The columns filled with numbers correspond to the number of events for a given status.
3. Choose a channel name (created in [Creating Communication Arrangements \[page 1256\]](#)) to access detailed information.  
The *Inbound Events* and *Outbound Events* tabs contain all inbound or outbound events by topic and status.
4. Select a *Topic*.  
All events for the given topic and status are listed.
5. Select an event to access detailed information.  
The detailed view of a consumer inbound event contains:

<i>Times</i>	<ul style="list-style-type: none"><li>• <i>Publish Time</i></li><li>• <i>Arrival Time</i></li><li>• <i>Consume Time</i></li><li>• <i>Fail Time</i></li></ul>
<i>Event Data</i>	<ul style="list-style-type: none"><li>• <i>Cloud Event Source</i></li><li>• <i>Cloud Event ID</i></li><li>• <i>Quality of Service</i></li></ul>
<i>Consumed by</i>	<ul style="list-style-type: none"><li>• <i>Consumer ID</i></li><li>• <i>Consumer Version</i></li></ul>
<i>Payload</i>	<ul style="list-style-type: none"><li>• <i>Payload</i></li><li>• <i>Size (kB)</i></li></ul>

Events reside in an internal queue of the enterprise event enablement. Once the processing of an event is finalized, it is deleted from the queue after a certain time period. Processing an event is finalized if the status is acknowledged or failed.

Deletion timeline	Event status	Description
Hourly before the current full hour	Acknowledged	The event has been successfully published or consumed according to the chosen quality of service. An hourly background job deletes all acknowledged events from the event monitor.
No deletion	Failed	The enterprise event enablement either failed to publish the event after 5 attempts or the event could not be consumed by the corresponding consumer. Failed events are stored until you either delete or retry them again.
No deletion	Not processed yet	All events that have a different status than acknowledged or failed are not deleted since they are not yet fully processed.

The timeline for deleting is subject to change.

## 4.2.10 SaaS Applications and Multitenancy

Find out how to develop and operate SaaS applications by using add-ons in the ABAP environment, see [Developing and Operating SaaS Applications \[page 1297\]](#).

Learn more about multitenancy in the ABAP environment, the multitenant application, tenant business types and lifecycle, and see the multitenancy development guideline: [Multitenancy in the ABAP Environment \[page 1407\]](#).

Find out how to use the [Landscape Portal](#), the central tool that allows providers to perform system administration tasks and lifecycle management operations.

Read about how SaaS applications can be extended, see [Extending SaaS Applications \[page 1420\]](#).

### 4.2.10.1 Developing and Operating SaaS Applications

Learn how to develop and operate SaaS applications by using add-ons in the ABAP environment.

#### Introduction

The ABAP environment allows you to leverage existing ABAP know-how to create custom add-ons in the cloud. This add-on build process is orchestrated by using the ABAP environment. Delivery tools, such as the Add-on Assembly Kit as a Service for the registration and publishing of the add-on as a software product, and deployment tools to carry out test installations and to make the add-on available for productive use, simplify this process. See [Delivery Tools](#) and [Deployment Tools](#).

#### ⓘ Note

You should only consider using add-on delivery if you intend to offer a Software as a Service (SaaS) solution.

#### ⓘ Note

Note that only one add-on can be installed per system.

Following the add-on build process, you can develop a multitenant application to share the add-on with multiple consumers (tenants) simultaneously. See [Multitenant Applications \[page 1376\]](#). The application is registered with the Software-as-a-Service provisioning service (saas-registry) to make the application available for subscription to consumers. See [Register the Multitenant Application to the SaaS Provisioning Service](#).

In this context, the [ABAP Solution Service \[page 1379\]](#) accounts for the provisioning of new ABAP service instances, including the installation of the add-on, and the creation of tenants whenever required for a new consumer subscription.

The way ABAP service instances and tenants are used for consumer subscription applications can be configured as follows:

- **Single-tenant enabled:** With each new consumer, a new ABAP service instance is created
- **Multi-tenant enabled:** For multiple consumers, the same ABAP service instance is used

For more information, see [Multitenancy in the ABAP Environment](#).

## What Is This Guide About?

This guide describes the end-to-end process of developing and offering a SaaS solution using the ABAP environment. The following chapters cover all aspects of this scenario, from the initial setup and development tasks to the final productive use by consumers.

The [Concepts \[page 1323\]](#) chapter gives you an overview about important concepts that are relevant throughout the end-to-end process.

The [Glossary \[page 1318\]](#) contains a list of all the technical terms and phrases used in this scenario.

Chapter [Develop, Test, Build \[page 1348\]](#) describes the actual development activities. This includes the setup of the required account structure and system landscape, as well as the recommended testing process and subsequent build of the add-on.

Chapter [Order and Provide \[page 1371\]](#) illustrates how a developed add-on can be deployed and made available to consumers. It also shows how consumers order solutions and what steps you, as the provider, need to perform to provision the solution.

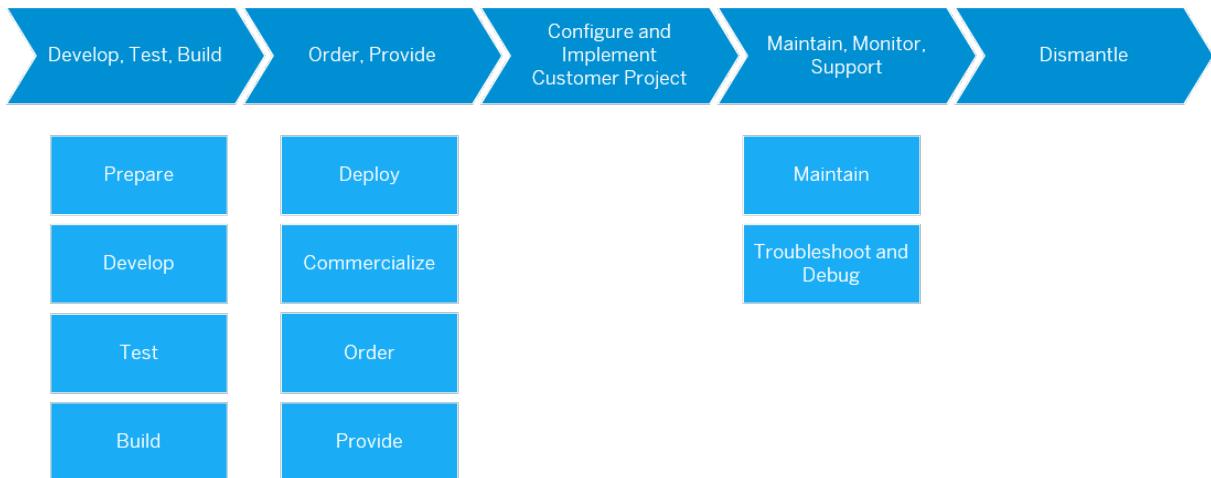
Chapter [Configure and Implement a Customer Project \[page 1395\]](#) describes how to configure and implement a customer project in the consumer tenant, including the setup of identity and authentication management, integration with other systems or services, and adjustment of business configuration.

Chapter [Maintain, Monitor, Support \[page 1398\]](#) gives you an overview about the activities involved in maintaining an already commercialized product, as you usually continue to support the add-on, fix bugs, and potentially offer new functionalities as time goes on.

Chapter [Dismantle \[page 1407\]](#) provides information about consumer offboarding, in particular about deleting and restoring tenants.

### Note

If you decide to use gCTS transport instead of add-on delivery, please refer to the gCTS Delivery boxes across the guide. See [Delivery via Add-On or gCTS \[page 1342\]](#).



- Prepare [page 1350]
- Develop [page 1361]
- Test [page 1364]
- Build [page 1367]
- Deploy [page 1371]
- Commercialize [page 1392]
- Subscribe [page 1393]
- Subscribe [page 1393]
- Maintain, Monitor, Support [page 1398]
- Order and Provide [page 1371]
- Develop, Test, Build [page 1348]
- Configure and Implement a Customer Project [page 1395]
- Maintain [page 1398]
- Troubleshoot and Debug [page 1405]
- Dismantle [page 1407]

For more information, see [Build ABAP Code into an Add-on Product and Make it Available as a Tenant-Aware SaaS Solution](#).

### **⚠ Caution**

The tutorial group "Convert Your Add-on Product into a Tenant-aware SaaS Solution" is now replaced by the functionality of the [Maintain Solution app](#).

## 4.2.10.1.1 Overview

### Develop, Test, Build

Phase	Step	Description	User/Role
<a href="#">Prepare [page 1350]</a>	Register a namespace. See <a href="#">Register a Namespace [page 1350]</a> .	A namespace is mandatory and must be reserved at <a href="https://support.sap.com/namespaces">https://support.sap.com/namespaces</a> .  <b>④ Note</b> You have to register a namespace before the first ABAP system is provisioned. To create namespaces after system provisioning, see <a href="#">Maintain Namespaces</a> .  Namespaces should have 5–8 characters due to length restrictions of certain objects. See SAP note <a href="#">105132</a> and <a href="#">395083</a> .	S-user

Phase	Step	Description	User/Role
	<p>Set up a global account for development. See <a href="#">Set Up a Global Account for Development [page 1351]</a>.</p> <p>Get a partner development license for discounted systems used for development, test, or demo purposes. These systems are provisioned in the global account for development.</p> <p>Required entitlements:</p> <ul style="list-style-type: none"> <li>• Services:           <ul style="list-style-type: none"> <li>• abap/standard and abap/ saas_oem</li> <li>• Application runtime for approuter</li> <li>• abap-solution (ABAP Solution Provider)</li> </ul> </li> <li>• Applications:           <ul style="list-style-type: none"> <li>• Landscape Portal</li> <li>• Web access for ABAP</li> <li>• SAP Business Application Studio for UI development</li> </ul> </li> </ul> <p>The saas-registry and xsuaa service are available without entitlement.</p> <p>See <a href="#">SAPPartnerEdge Test, Demo and Development Price List</a>.</p>		Operator

Phase	Step	Description	User/Role
	<p>Set up a global account for production. See <a href="#">Set Up a Global Account for Production [page 1354]</a>.</p> <p>Get a partner production license for systems used for production purposes. These systems are provisioned in the global account for production.</p> <p>Required entitlements:</p> <ul style="list-style-type: none"> <li>• Services: <ul style="list-style-type: none"> <li>• abap/saas_oem</li> <li>• Application runtime for approuter</li> <li>• abap-solution (ABAP Solution Provider)</li> <li>• saas-registry (without entitlement)</li> <li>• xsuaa (without entitlement)</li> </ul> </li> <li>• Applications: <ul style="list-style-type: none"> <li>• Landscape Portal</li> <li>• Web access for ABAP</li> </ul> </li> </ul> <p>See <a href="#">SAPPartnerEdge: Resources for OEM Partners</a>.</p>		Operator

Phase	Step	Description	User/Role
	Create ABAP instance(s). See <a href="#">Create ABAP Instances [page 1355]</a> .	<p>Set up your development/test subaccounts and ABAP system and create the following:</p> <ul style="list-style-type: none"> <li>• Development system DEV with parameter <code>is_development_allowed = true</code> <ul style="list-style-type: none"> <li>• abap/abap_compute_unit (standard: 1)</li> <li>• abap/hana_compute_unit (standard: 4)</li> </ul> </li> <li>• Test system TST with parameter <code>is_development_allowed = false</code> <ul style="list-style-type: none"> <li>• abap/abap_compute_unit (standard: 1)</li> <li>• abap/hana_compute_unit (standard: 4)</li> </ul> </li> <li>• Subscriptions in development/test subaccount to ABAP Web Access</li> </ul> <p>See <a href="#">Getting Started with a Customer Account: Workflow in the ABAP Environment</a> and <a href="#">Create an ABAP System</a>.</p>	Operator
	Set up UI development. See <a href="#">Set Up UI Development [page 1356]</a> .	Configure SAP Business Application Studio for UI development.	Operator
	Set up add-on development. See <a href="#">Set Up Add-On Development [page 1357]</a> .	Create a new software component for add-on development and clone the main branch of the software component to the development and test system.	Add-on administrator

Phase	Step	Description	User/Role
	Transport from development to test system. See <a href="#">Set Up Transport from Development to Test System [page 1358]</a> .	Plan and set up the transport of software components from development to test systems.  Tutorial: <a href="#">Transport a Software Component Between two Systems</a>	Add-on administrator
		<p><b> ⓘ Note</b></p> <p>A CI/CD server that is running the ABAP Environment Pipeline is required. See <a href="#">ABAP Environment Pipeline [page 1328]</a>.</p>	
Develop [page 1361]	ABAP development. See <a href="#">ABAP Development [page 1361]</a> .	Follow the guidelines to enable multitenancy for applications built on the ABAP environment.  See <a href="#">Multitenancy Development Guideline [page 1412]</a> and <a href="#">Tutorials</a> .	Developer
	UI development. See <a href="#">UI Development [page 1363]</a> .	Develop UIs using SAP Business Application Studio. See <a href="#">Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio [page 1841]</a> .	Developer
	[Optional] Code migration. See <a href="#">(Optional) Code Migration from On-Premise [page 1364]</a> .	Migrate your ABAP code from on-premise.  See <a href="#">Blog: How to bring your ABAP custom to the ABAP Environment</a> and <a href="#">Blog: How to check your custom ABAP code for the ABAP Environment</a> .	
Test [page 1364]	Test in cloud system. See <a href="#">Test in the ABAP Environment SAP Fiori Launchpad [page 1365]</a> .	Import the ABAP service and the UI into your test system.  Create a role and assign it to users, then test the service and UI.	Test user

Phase	Step	Description	User/Role
	<p>Test in the ABAP Test Cockpit. See <a href="#">Test in the ABAP Test Cockpit [page 1366]</a>.</p> <p>See <a href="#">ABAP Test Cockpit in the Cloud – What is already possible</a> and <a href="#">ABAP Test Cockpit Configurator [page 2553]</a>.</p>	<p>Run ABAP Test Cockpit checks in DEV and/or TST systems.</p>	Developer
Build <a href="#">[page 1367]</a>	<p>Prepare your add-on build. See <a href="#">Set Up Add-On Build [page 1368]</a>.</p> <p>Build your first add-on version. See <a href="#">Build the First Add-On Version [page 1370]</a>.</p>	<p>Set up a CI/CD server and pipeline for the add-on build.</p> <p>Create a technical communication user to access AA-KaaS.</p> <p>See <a href="#">Build and Publish Add-on Products on SAP BTP, ABAP Environment</a>.</p> <p>Register the add-on product in the global account for development and production as described in <a href="#">Register Add-on Product for a Global Account</a> and <a href="#">Register Product</a>.</p>	<ul style="list-style-type: none"> <li>Jenkins administrator for access to CI/CD Server</li> <li>S-user to create technical communication user</li> <li>Operator to assign technical platform user as space developer</li> <li>DevOps engineer to configure the pipeline</li> </ul>

## Order and Provide

Phase	Step	Description	User/Role
<a href="#">Deploy [page 1371]</a>	Configure the sizing of the SaaS application. See <a href="#">Sizing [page 1372]</a> .	<p>Decide on the metric of your offering (for example users, documents, space) and map the metric of your offering to the service plan of the ABAP environment service.</p> <p><b> ⓘ Note</b></p> <p>For multitenancy offerings, there is no sizing/quota per customer (client). You must decide on an overall sizing depending on the expected load in a region (for example Europe/Frankfurt).</p> <p>As a DevOps engineer using parameter <code>tenant_mode</code> in the ABAP Solution service, you can define whether a customer gets a tenant in a dedicated system (single) or a shared system (multi).</p> <p>Determine the number of ABAP compute units and HANA compute units for the creation of an ABAP system by using parameters <code>size_of_runtime</code> and <code>size_of_persistence</code>.</p>	DevOps engineer
	Implement a multitenant application and deploy it to the provider subaccount in the global account for development. See <a href="#">Multitenant Applications [page 1376]</a> .	Create the multitenant application with SaaS registry, XSUAA, and ABAP Solution Provider. Deploy to the provider subaccount with parameters used for development phase.	DevOps engineer

Phase	Step	Description	User/Role
Deploy [page 1391]	Access the Landscape Portal. See <a href="#">Access to Landscape Portal [page 1349]</a> .	Subscribe to the Landscape Portal application in the global accounts for development and production.  See <a href="#">Accessing the Landscape Portal</a> .	Operator
	Test the multitenant application deployed to the global account for development.	Subscribe to the SaaS solution from the consumer subaccount.	DevOps engineer
	Deploy the multitenant application to the provider subaccount in the global account for production.	Deploy to the provider subaccount with parameters used for production phase.	DevOps engineer
Commercialize [page 1392]	[Optional] SAP Store registration. See <a href="#">(Optional) Register SaaS Solution in SAP Store [page 1392]</a> .	Register your SaaS solution in SAP Store.  See <a href="https://store.sap.com/dcp/en/">https://store.sap.com/dcp/en/</a> .	
	[Optional] Certification. See <a href="#">(Optional) Certification [page 1393]</a> .	Get an optional certification for your product.  See <a href="#">Certification</a> and <a href="#">Certified Solutions</a> .	
Subscribe [page 1393]	Create a consumer subaccount. See <a href="#">Create Consumer Subaccount [page 1393]</a> .	Set the account structure for the consumer subaccount.	Operator
	Subscribe to the SaaS solution. See <a href="#">Subscribe to SaaS Solution [page 1393]</a> .		Operator
	Configure consumer subaccount. See <a href="#">Configure Consumer Subaccount [page 1394]</a> .	Configure customer subaccount.  <ul style="list-style-type: none"> <li>• Option A: provider configures the subaccount</li> <li>• Option B: customer configures subaccount</li> </ul>	Consumer subaccount administrator
	Initial user onboarding. See <a href="#">Initial User Onboarding [page 1395]</a>	Create an initial administrator user for the customer.	Consumer subaccount administrator

## Configure and Implement a Customer Project

Phase	Step	Description	User/Role
Configure and Implement a Customer Project [page 1395]	Set up Identity and Access Management in the consumer tenant of the customer.	Create and assign business roles. Create spaces and pages for business roles.	Business user
	Integrate the customer project in the consumer tenant of the customer.	Create communication arrangements.	Business user
	Configure the customer project in the consumer tenant of the customer.	Create business configuration. Configure key user extensibility.	Business user

## Maintain, Monitor, Support

Phase	Step	Description	User/Role
Maintain [page 1398]	Set up your test and maintenance system landscape and processes. See <a href="#">Set Up Maintenance System Landscape [page 1399]</a> .		Operator
	Create an update. See <a href="#">Create Add-On Update [page 1400]</a> .	Create a new patch version, support package version, or release/product version. See <a href="#">Deploy Product</a> .	<ul style="list-style-type: none"> <li>• Developer to implement update and double-maintenance</li> <li>• Test user to verify changes in test systems</li> <li>• Add-on administrator to check out maintenance branch, import software components into test systems, configure add-on.yml file</li> </ul>
	Trigger the add-on product build. See <a href="#">Trigger Add-On Build Pipeline [page 1403]</a> .	Trigger the execution of the configured ABAP environment pipeline for an add-on build. See <a href="#">Operations Dashboard</a> .	Add-on administrator
	Apply update for SaaS solution (=add-on product). See <a href="#">Deploy Add-On Update [page 1404]</a> .	Deploy your add-on update by using the Landscape Portal.  See <a href="#">Landscape Portal</a> .	Operator

Phase	Step	Description	User/Role
Troubleshoot and debug. See <a href="#">Troubleshoot and Debug [page 1405]</a> .	<ul style="list-style-type: none"> <li>Identify ABAP system and consumer tenant</li> <li>Access the consumer tenant of the ABAP system</li> <li>Troubleshoot</li> </ul>		Operator

## Dismantle

Phase	Step	Description	User/Role
<a href="#">Dismantle [page 1407]</a>	Delete tenants.	Unsubscribe the SaaS solution. See <a href="#">Dismantle [page 1407]</a> .	Consumer subaccount administrator or operator using Subscription Management Dashboard
	(Optional) Restore deleted tenants.	Restore the deleted tenants within a grace period of 30 days. See <a href="#">Dismantle [page 1407]</a> .	Operator

## 4.2.10.1.2 Users and Roles

User	Role	Description	Tasks/Usage
Provider	Add-On Administrator	The add-on administrator is responsible for everything related to the add-on production, for example, releasing an add-on product version.	<ul style="list-style-type: none"> <li>Creating software component</li> <li>Importing changes in software component to test systems</li> <li>Adjusting add-on descriptor file <code>addon.yml</code> for new versions</li> <li>Registering add-on product for installation in global account for development and production. See <a href="#">Register Product</a>.</li> <li>Executing add-on build pipeline in Jenkins instance</li> <li>Confirming Integration Tests stage in add-on build pipeline</li> <li>Trigger publishing in Confirm stage during add-on build pipeline</li> <li>Creating maintenance branches (for example v1.0.0) per support package level in <a href="#">Manage Software Components</a> app. See <a href="#">Manage Software Components [page 2805]</a>.</li> <li>Checking out maintenance branch in correction system COR and quality assurance system QAS using the <a href="#">Manage Software Components</a> app.</li> </ul>

User	Role	Description	Tasks/Usage
	DevOps Engineer	The DevOps engineer is responsible for the configuration of the pipeline and implementation of the multitenant application that needs to be deployed to enable the add-on as a SaaS solution.	<ul style="list-style-type: none"> <li>Initial configuration of add-on pipeline, especially <code>config.yml</code> file in git repository</li> <li>Implementing pipeline extensions if necessary</li> <li>Implementing a multitenant application. See <a href="#">Multitenant Applications [page 1376]</a>.</li> <li>Deploy multitenant application to provider subaccount in global account for development</li> <li>Subscribe to multitenant application from consumer subaccount in global account for development</li> <li>Deploy multitenant application to provider subaccount in global account for production</li> </ul>

User	Role	Description	Tasks/Usage
	Developer	<p>Developer users use ABAP Development Tools (ADT) to create backend service artifacts and UI developers create and deploy SAP Fiori apps in SAP Business Application Studio. See <a href="#">What is SAP Business Application Studio</a>.</p>	<ul style="list-style-type: none"> <li>Developing backend services in ABAP Development Tools (ADT) using ABAP RESTful Application Programming Model. See <a href="#">Eclipse Tool for the ABAP Environment</a> and <a href="#">ABAP RESTful Application Programming Model [page 71]</a>.</li> <li>Developing SAP Fiori based UI in SAP Business Application Studio and deploying back to ABAP system. See <a href="#">Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio [page 1841]</a>.</li> <li>Supporting and troubleshooting customer issues via provider support access in the Landscape Portal. See <a href="#">Landscape Portal</a>.</li> </ul>
	Test User	<p>Test users are business users in test systems TST and QAS that validate the correct implementation of the add-on. This involves, among others, maintaining business roles and communication arrangements delivered as part of the add-on.</p>	<ul style="list-style-type: none"> <li>Maintaining business roles. See <a href="#">Maintain Business Roles</a>.</li> <li>Creating spaces and pages for business roles. See <a href="#">How to Create and Assign Spaces and Pages</a>.</li> <li>Maintaining communication arrangements. See <a href="#">Communication Arrangements</a>.</li> <li>Creating business configuration</li> <li>Creating key user extensibility</li> <li>Testing ABAP backend services</li> <li>Testing SAP Fiori UIs</li> </ul>

User	Role	Description	Tasks/Usage
	Provider Identity Authentication Administrator	An Identity Authentication service tenant can be configured for authentication in development/test/assembly systems. The Identity Authentication administrator configures the trust setup between the subaccount and Identity Authentication application. See <a href="#">SAP Cloud Identity Services - Identity Authentication</a> .	<ul style="list-style-type: none"> <li>Configuring Identity Authentication application for trust setup in development/test/assembly subaccounts. See <a href="#">Create New Application</a>.</li> <li>Configuring corporate identity provider in Identity Authentication tenant. See <a href="#">Corporate Identity Providers</a>.</li> </ul>

User	Role	Description	Tasks/Usage
	Operator	<p>The SaaS solution operator is responsible for creating the account model on the provider side (creating consumer subaccounts), adding subaccount members etc.</p>	<ul style="list-style-type: none"> <li>Setting up global accounts for development and production including entitlements, subaccounts, Cloud Foundry org, spaces, services, and apps, as well as trust configuration and connectivity per subaccount. See <a href="#">System Landscape/Account Model [page 1323]</a>.</li> <li>Consumer tenant onboarding: creating consumer subaccount and subscription. See <a href="#">Subscribe to Multitenant Applications Using the Cockpit</a>.</li> <li>Creating users in ABAP systems. See <a href="#">Creation of Additional Administrator Users</a>. <ul style="list-style-type: none"> <li>Developer users in DEV/COR systems. See <a href="#">Creation of Developer Users</a>.</li> <li>Tester users in TST/QAS systems</li> </ul> </li> <li>Add-on update of existing systems in the Update Product Version app. See <a href="#">Deploy Product</a>.</li> <li>Monitoring of system/tenant provisioning and user onboarding. See <a href="#">Operations Dashboard</a>.</li> <li> <b> ⓘ Note</b>  If you use gCTS for delivery to customer production systems instead of using add-ons, you </li> </ul>

User	Role	Description	Tasks/Usage
			<p>are responsible for cloning software components and pulling the latest changes into the system using the <a href="#">Manage Software Components</a> app.</p> <p>See <a href="#">Delivery via Add-On or gCTS [page 1342]</a>.</p>
Jenkins Administrator		<p>A Jenkins CI/CD server is used as the infrastructure for automation purposes. Pipeline configuration, user management, credentials management etc. is handled by the Jenkins administrator. See <a href="#">Continuous Integration and Delivery (CI/CD) [page 125]</a>.</p>	<ul style="list-style-type: none"> <li>• Creating Jenkins instance via Cx server</li> <li>• Maintaining Jenkins credentials for: <ul style="list-style-type: none"> <li>• Cloud Foundry technical platform user</li> <li>• Technical communication user</li> <li>• Credentials of service user for git repository of pipeline configuration</li> </ul> </li> <li>• Maintaining Jenkins pipeline with reference to git repository of pipeline configuration</li> <li>• Adding additional Jenkins users, for example for add-on administrator user</li> </ul>

User	Role	Description	Tasks/Usage
	S-User	<p>S-users are used by partners and customers to log on to SAP ONE Support Launchpad to reserve development namespaces or to create technical communication users. User management is taken care of by user administrators in the <a href="#">User Management</a> app. If customer numbers are assigned to a Partner ID, S-users are registered via PartnerEdge. See <a href="#">User Management</a> and SAP note <a href="#">2261006</a>.</p>	<ul style="list-style-type: none"> <li>Reserving development namespace in Namespace Application See <a href="https://launchpad.support.sap.com/#/namespaces">https://launchpad.support.sap.com/#/namespaces</a>.</li> <li>Activating new namespaces in already existing systems using Landscape Portal. See <a href="#">Maintain Namespaces</a>.</li> <li>Creating technical communication user. See SAP note <a href="#">2532813</a>.</li> </ul>
Technical Users of Provider	Technical Cloud Foundry Platform User	<p>A platform user is added as a member to subaccounts in the Cloud Foundry environment to create or delete ABAP service instances, create service keys etc. This user should be a non-personalized user, for example based on a distribution list. The user does not need authorizations in SAP ONE Support Launchpad. The user does not need authorizations in SAP ONE Support Launchpad. Platform users are created by default in SAP ID Service. See <a href="#">User and Member Management</a> and <a href="#">Default Identity Provider</a>.</p>	<ul style="list-style-type: none"> <li>Creating ABAP system AMT via ABAP solution service for consumer tenants. See <a href="#">Define Your ABAP Solution</a>.</li> <li>Creating ABAP system BLD via add-on build pipeline for add-on assembly</li> <li>Creating ABAP system ATI via add-on build pipeline for add-on installation test</li> <li>Deleting ABAP systems</li> </ul>
	Service User Git	<p>Jenkins pipeline definitions are source-code-based and stored in a source code repository, usually a Git repository. To retrieve the pipeline definition from the repository, a (technical) service user is configured.</p>	<ul style="list-style-type: none"> <li>Pulling changes in git repository of pipeline configuration</li> <li>Pulling changes in SAP/Jenkins library to project "Piper" library steps</li> </ul>

User	Role	Description	Tasks/Usage
	Technical Communication User	A technical communication user (created via SAP ONE Support Launchpad) is used for system-to-system connections with the SAP Support backbone. For information on how technical communication users are different from S-users, see SAP note <a href="#">2668288</a> .	Used for communication with the AAKaaS. See <a href="#">Add-on Assembly Kit as a Service (=AAKaaS)</a> .
Consumer	Consumer Identity Authentication Administrator	An Identity Authentication Service tenant can be configured for authentication at consumer tenants in the productive ABAP systems. The Identity Authentication admin configures the trust setup between consumer subaccount and Identity Authentication application. See <a href="#">Identity Authentication Service</a> .	<ul style="list-style-type: none"> <li>Configuring Identity Authentication application for trust setup in consumer subaccount</li> <li>Optional: Configuring corporate identity provider in Identity Authentication tenant</li> </ul>
	Consumer Subaccount Administrator	The consumer subaccount administrator is responsible for the configuration of: <ul style="list-style-type: none"> <li>Trust settings (custom identity provider), see <a href="#">Trust and Federation with Identity Providers</a>.</li> <li>Connectivity via Cloud Connector, see <a href="#">Connectivity in the Cloud Foundry Environment</a></li> <li>Destinations, see <a href="#">Managing Destinations</a> in the consumer subaccount created for the customer.</li> </ul>	<ul style="list-style-type: none"> <li>Setting up connectivity via Cloud Connector</li> <li>Adding destinations in consumer subaccount</li> <li>Setting up trust configuration</li> </ul>

User	Role	Description	Tasks/Usage
	Business User	Business users make use of the SaaS solution in the consumer tenant for their daily work.	<ul style="list-style-type: none"> <li>Initial administrator on-boarding</li> <li>Creating additional business users</li> <li>Creating business roles</li> <li>Creating spaces and pages for business roles. See <a href="#">How to Create and Assign Spaces and Pages</a>.</li> <li>Creating communication arrangements</li> <li>Creating business configuration</li> <li>Creating key user extensibility</li> <li>Using SAP Fiori UIs</li> </ul>
	Cloud Connector Administrator	To connect on-premise systems on the customer side to the consumer tenant with the SaaS solution, Cloud Connector can be configured in the consumer subaccount. The Cloud Connector administrator maintains the consumer subaccount, cloud-to-on-premise system mappings etc. See <a href="#">Cloud Connector</a> .	<ul style="list-style-type: none"> <li>Adding a consumer subaccount to Cloud Connector. See <a href="#">Managing Subaccounts</a>.</li> <li>Configuring cloud to on-premise system mappings. See <a href="#">Configure Access Control</a>.</li> <li>Synchronizing trust configuration for principal propagation. See <a href="#">Set Up Trust for Principal Propagation</a>.</li> </ul>

### 4.2.10.1.3 Glossary

Add-on Assembly Kit as a Service (AAKaaS)

The Add-on Assembly Kit as a Service registers and publishes the software product. It is accessible via APIs with a technical communication user.

ABAP environment pipeline

The ABAP environment pipeline enables continuous integration for the ABAP environment. The pipeline contains several stages and supports different scenarios. See [ABAP Environment Pipeline](#).

ABAP System	The ABAP environment service (abap/standard) is used to create development and test systems. See <a href="#">Creating an ABAP System</a> .
ABAP System (SaaS, OEM)	The ABAP environment (abap/saas_oem) service is used to create the add-on installation test system and production systems, where a specific add-on product needs to be installed in the system.
Add-on build pipeline	A pipeline configured for the add-on build scenario of the ABAP environment pipeline. See <a href="#">Build and Publish Add-on Products on SAP BTP ABAP Environment</a> .
Add-on descriptor	The build process is controlled by an add-on descriptor file called addon.yml. This file must be created manually and stored in the Git repository of the pipeline. It must contain information about the software product version to be delivered and the containing software component versions. See <a href="#">Add-On Descriptor File</a> .
Add-on installation test system	To verify that the delivery packages included in the add-on product version being built are installable, a target vector is published in "test" scope. In the integration tests stage, an ABAP system of service plan saas_oem is created. This ABAP OEM service allows you to install a specific add-on product version into an ABAP system that is provisioned.
Add-on product version (software product version)	To install and maintain ABAP software, software product versions are used. A software product version is a bundle of software component versions made available at the same time for implementing a well-defined scope of functionality. See <a href="#">Software Product Version</a> .
ABAP Solution (Provider)	The ABAP Solution service maintains ABAP systems and tenants to provide an add-on product as a SaaS solution.
Assembly System	The ABAP system responsible for the add-on build.
ABAP Test Cockpit	ABAP Test Cockpit checks can be executed using abapEnvironmentRunATCCheck. The step can receive software components or packages configured in a YML file. The results are returned in checkstyle format. With the use of a pipeline extension, you can configure quality gates. See <a href="#">ATC</a> and <a href="#">ABAP Test Cockpit Configurator [page 253]</a> .
Branch	Git repository branches can be used to control the flow of changes through the test and production landscape. They separate changes from each other, which shall or might not be delivered together (i.e. development vs. correction, feature A vs. feature B). A branch is created on the current state of another branch, the parent branch, and reflected by a new copy of all included objects. Depending on which branch is supposed to be changed, the corresponding copy is worked on.

Business role	A business role provides users with authorizations to access apps.
CI/CD server	The pipeline for building ABAP add-ons has been created specifically for Jenkins. Therefore, a Jenkins server is required. The project "Piper" provides a Jenkins image, which already includes the necessary configurations. Note that you can also configure an existing server.
Cloud Foundry environment	The Cloud Foundry environment enables you to develop new business applications and business services, supporting multiple runtimes, programming languages, libraries, and services. See <a href="#">Cloud Foundry Environment [page 49]</a> .
Communication arrangement	A communication arrangement describes a communication scenario with a remote system during configuration time. It provides the necessary metadata for service configuration. See <a href="#">Communication Arrangement [page 885]</a> .
Communication scenario	A communication scenario in the ABAP environment is a design time description of how two communication partners communicate with each other.
Consumer	It provides technical information, such as the used inbound and outbound services and their service type, for example OData or SOAP, and the number of allowed communication arrangement instances. See <a href="#">Communication Scenario [page 883]</a> .
Correction codeline	The service consumer is an end-customer of the service provider and has access to the SaaS application in a specific tenant
Cx server	Corrections need to run in parallel to development and on a released state. This is performed in the correction codeline and on a maintenance branch.
Destination service	Cx server is a life-cycle management tool to bootstrap a preconfigured Jenkins instance within minutes on your own (virtual) server, that uses Docker images. To avoid manual startup of the Docker image with all the required parameters and sidecar images, this command line tool automates the bootstrapping. See <a href="#">Cx Server</a> .
Development codeline	The Destination service lets you find the destination information that is required to access a remote service or system from your Cloud Foundry application. See <a href="#">Consuming the Destination Service</a> .
Development namespace	Infinity development (infinity codeline) is done based on the latest software component state in the main branch.

Global account	A global account is the realization of a contract you made with SAP. A global account is used to manage subaccounts, members, entitlements, and quotas. You receive entitlements and quotas to use platform resources per global account and then distribute the entitlements and quotas to the subaccount for actual consumption. See <a href="#">Getting a Global Account [page 144]</a> .
Identity Provider	SAP Business Technology Platform supports identity federation, a concept of linking and reusing digital identities of a user base across loosely coupled systems. Identity federation frees applications on SAP Business Technology Platform from the need to obtain and store the credentials of users to authenticate them. Instead, the application user base is reused from identity providers, which support the administration of digital user identities, authentication, and authorizations in a centralized and decoupled manner. See <a href="#">Trust and Federation with Identity Providers [page 2204]</a> .
Landscape Portal	The Landscape Portal app acts as a central tool to allow service providers to perform lifecycle management operations such as add-on updates, provisioning new consumers as new tenants, and more. See <a href="#">Landscape Portal</a> .
Multitenancy	A multitenant service/application serves requests from different customers - the tenants - and processes their data strictly isolated from one another.
Patch level	Patch deliveries shall only contain small corrections. They are shipped with delivery packages of type <a href="#">Correction Package</a> .
Project "Piper"	For both the add-on product version and the software component version, the patch level is denoted in the third digit.  SAP implements tooling for continuous delivery in project "Piper". The goal of project "Piper" is to substantially ease setting up continuous delivery in your project using SAP technologies. See <a href="#">Project Piper</a> .
Provider	The provider is responsible for the development and maintenance of the SaaS application. This is typically an independent software vendor or development partner.
Release	Release deliveries contain the whole software component and deliver new features and enhancements of existing functionalities.  For both the add-on product version and the software component version, the release is denoted in the first digit.

SAP Business Application Studio	Available as a cloud service, SAP Business Application Studio provides a desktop-like experience similar to leading IDEs, with command line and optimized editors. At the heart of SAP Business Application Studio are the dev spaces, which are similar to isolated virtual machines in the cloud containing tailored tools and pre-installed runtimes per business scenario, such as SAP Fiori, SAP S/4HANA extensions, Workflow, Mobile and more. This simplifies and saves time in setting up your development environment and allows you to efficiently develop, test, build, and run your solution locally or in the cloud. See <a href="#">What is SAP Business Application Studio</a> .
SAP Cloud Identity Services - Identity Authentication	The Identity Authentication service provides you with controlled cloud-based access to business processes, applications, and data. It simplifies your user experience through authentication mechanisms, single sign-on, on-premise integration, and convenient self-service options. See <a href="#">What Is Identity Authentication?</a> .
SAP Cloud Identity Services - Identity Provisioning	The Identity Provisioning service automates identity lifecycle processes. It helps you provision identities and their authorizations to various cloud and on-premise business applications. See <a href="#">What is Identity Provisioning</a> .
SAP ID service	The default platform identity provider and application identity provider of SAP Business Technology Platform is SAP ID service. See <a href="#">Default Identity Provider [page 2258]</a> .
SAP ONE Support launchpad	SAP ONE Support Launchpad provides you access to task-driven support resources in an intuitive interface. By using customizable role profiles, it displays only the relevant applications and insights to you. See <a href="#">SAP ONE Support Launchpad</a> .
SAP PartnerEdge	Get information, training, tools, and resources regarding partner licensing at <a href="#">SAP Partner Portal</a> .
SAP Store	SAP Store is the enterprise marketplace where we bring together customers and partners on a single, easy-to-use, global online platform. Here, customers can discover, try, and buy SAP-validated partner applications that are built on or extend their existing SAP technology and solutions. For partners, it's the only place they can market and deliver their apps, add-ons, and integration kits to SAP's global customers — solutions that help customers grow their business. See <a href="#">SAP Store</a> .
Software component	In ABAP environment systems, you develop within software components (also called repositories). The add-ons being built in this scenario are made up by one or multiple software components combined to an add-on product. See <a href="#">Software Components [page 717]</a> .
Software component version	A software component version is a technically distinguishable unit of software and is installed and patched as a whole. See <a href="#">Software Component Version</a> .

Subaccount	Subaccounts allow you to structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements. See <a href="#">Subaccounts</a> .
Support Package (Stack) Level	Support package deliveries contain a larger collection of corrections and may contain smaller functional enhancements. For both the add-on product version and the software component version, the support package level is denoted in the second digit.
Web Access for ABAP	Subscribe to the Web access for ABAP to get direct browser access to your instances in the ABAP environment, including access to the administration launchpad for ABAP. See <a href="#">Subscribing to the Web Access for ABAP [page 186]</a> .

## 4.2.10.1.4 Concepts

Learn more about the system landscape/account model, ABAP environment pipeline, as well as versioning and branches.

### 4.2.10.1.4.1 System Landscape/Account Model

#### ⓘ Note

If you use gCTS transport delivery instead of add-on delivery, no discounted development licenses are available: ABAP systems created for development purposes (development, test, demo) cost the same as ABAP systems created for production purposes. For creation on system AMT instead of `saas_oem`, the standard service plan is used since no add-on is installed.

Additionally, the same global account is used for development/production purposes because gCTS repositories are only available across the same global account.

ABAP systems for development and for production purposes are created based on different partner contracts and licenses.

Discounted development licenses can be used for development, test, and demo purposes.

#### ⓘ Note

You have to acquire a development license for partners. See [SAP PartnerEdge Test, Demo & Development Price List](#).

Production licenses are used whenever one of your customers is consuming the solution provided by you in one of your systems.

#### ⓘ Note

You have to acquire a production license for partners. See [Resources for OEM Partners](#).

To separate development and production purposes, you have to create different global accounts:

- **Global account for development**

The global account for development is used for add-on/UI development and testing. Also, add-on assembly and add-on test installation triggered by the add-on build pipeline are performed in this global account.

All ABAP systems created in this global account are used for development, test, or demo purposes.

- **Global account for production**

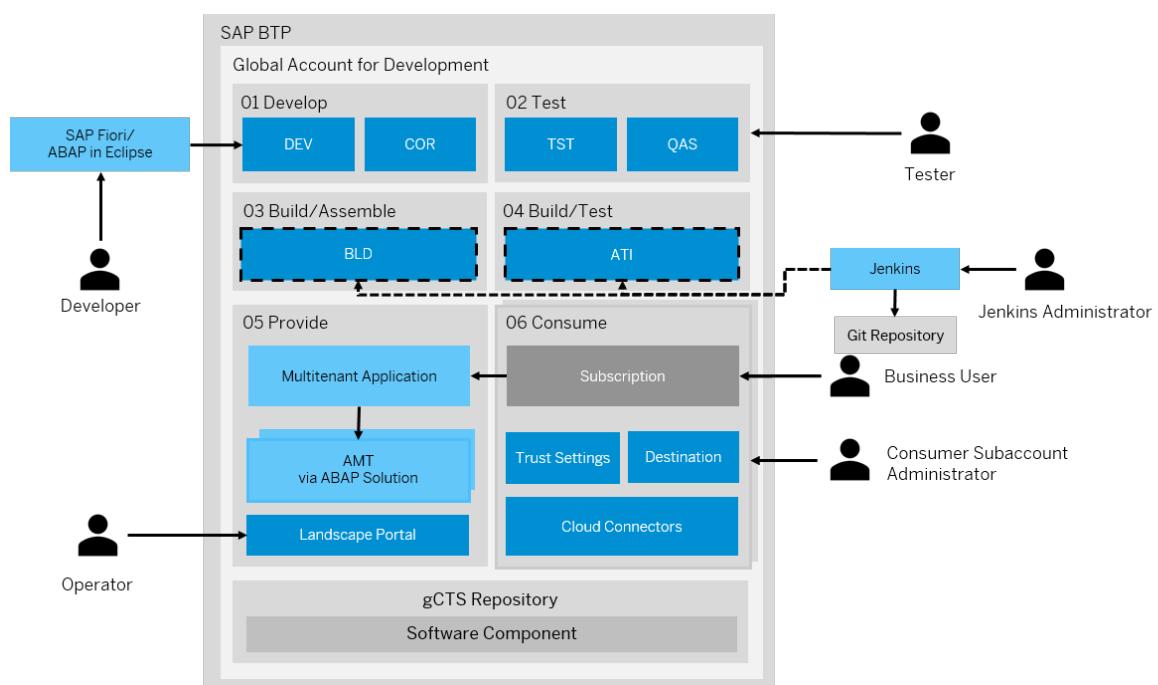
All ABAP systems created in this global account are used for production purposes. That means, the systems are used to provide the SaaS solution to your customers. Consumer subaccounts are also created in the global account for production.

### → Recommendation

We recommend creating subaccounts for the different stages of the SaaS solution enablement.

Using different subaccounts for the different phases has multiple advantages: Trust settings and destination/connectivity settings can be adjusted for each subaccount. That means, you can connect the development subaccount to the developer user identity provider and the test subaccount to the test user identity provider. In terms of connectivity, the development subaccount can be connected to the development on-premise system and the test subaccount can be connected to the test on-premise system.

## Global Account for Development



### 1. 00 Landscape Portal

Since the Landscape Portal service is only available in region eu10, we recommend creating a dedicated subaccount in this region. This subaccount will contain your Landscape Portal & CI/CD service subscriptions for you to freely choose other regions for your other subaccounts.

### Note

If you plan to create any of the other subaccounts in eu10, then you can simply reuse one of those subaccounts.

## 2. 01 Develop: Development subaccount including development space

ABAP development system DEV and correction system COR are created in this subaccount. You need to subscribe to SAP Business Application Studio to develop SAP Fiori UIs and implement multitenant applications. See [Setup of UI Development in SAP Business Application Studio](#).

## 3. 02 Test: Test subaccount including test space

ABAP test system TST and quality assurance system QAS are created in this subaccount. See [Use Case 2: One Development and Correction Codeline in a 5-System Landscape \[page 747\]](#). Since development is structured with software components that are stored in a repository for each global account, these software components can automatically be imported to the test system on a regular basis automated by the CI/CD server. See [\(Deprecated\) Test Integration \(SAP\\_COM\\_0510\) \[page 785\]](#). The CI/CD server uses a Git repository to read the pipeline definition and configuration.

## 4. 03 Build/Assemble: Subaccount for add-on assembly including build/assemble space

### Note

The ABAP environment platform version of the assembly system is used to determine the minimum platform version for the add-on product version that is created.

Such a system should not be nominated for the pre-upgrade option of the ABAP environment because this would lead to the add-on product only being able to installed in systems with the pre-upgrade release. When using Build Product Version (recommended), the add-on definition is configured directly in the app.

For the add-on build process, an assembly system BLD is provisioned in this subaccount by the CI/CD server. See [Software Assembly Integration \(SAP\\_COM\\_0582\) \[page 795\]](#). When using the Build Product Version app (recommended), the add-on definition is configured directly in the app.

## 5. 04 Build/Test: Subaccount for add-on installation test including build/test space

After the add-on has been assembled during the build of an add-on version, an installation test is required to verify that the add-on can be installed without errors into a system. As part of the add-on build pipeline, an abap/saas\_oem system ATI is provisioned in this subaccount and the add-on is installed. When using the Build Product Version app (recommended), the add-on definition is configured directly in the app.

## 6. 05 Provide: Provider subaccount including provider space

During the development phase, the multitenant application is deployed to this space for testing purposes. The ABAP solution service can then provision abap/saas\_oem systems AMT, tenants, and users in this account, once a consumer subscribes to the provided SaaS solution.

## 7. 06 Consume: Consumer subaccount

To test the subscription to the SaaS solution during the development phase of the multitenant application, a consumer subaccount is created in the global account for development. This allows your consumers to have their own configuration of:

- Trust settings (custom identity provider)

### Note

If you want to integrate an existing corporate identity provider for authentication/authorization in subaccounts of the global account for development, see [Trust and Federation with Identity](#)

**Providers.** To restrict access based on certain criteria, such as the IP address, you need to use the Identity Authentication service. See [SAP Cloud Identity Services - Identity Authentication](#).

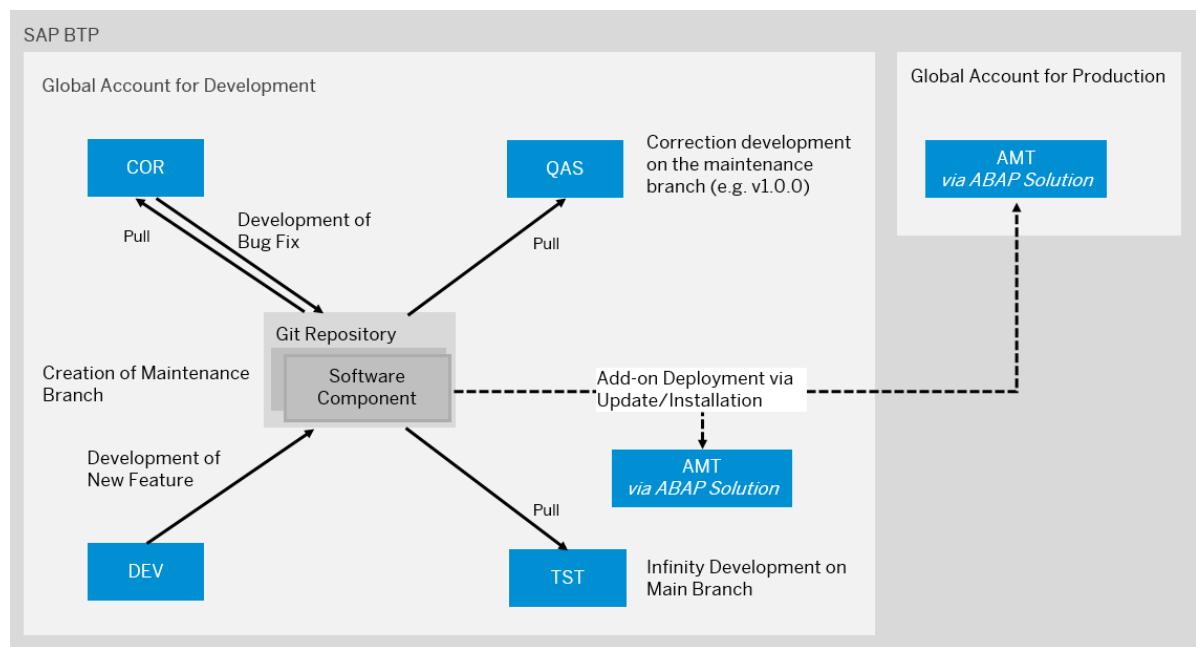
- Connectivity via SAP Cloud Connector. See [Connectivity in the Cloud Foundry Environment](#).
- Destinations. See [Managing Destinations](#).
- Subscriptions

### ⓘ Note

You can use a booster (see [Booster for Landscape Portal \[page 1358\]](#)) to automatically perform the setup of subaccounts 00 Landscape Portal, 03 Build/Assemble and 04 Build/Test.

## Development Flow

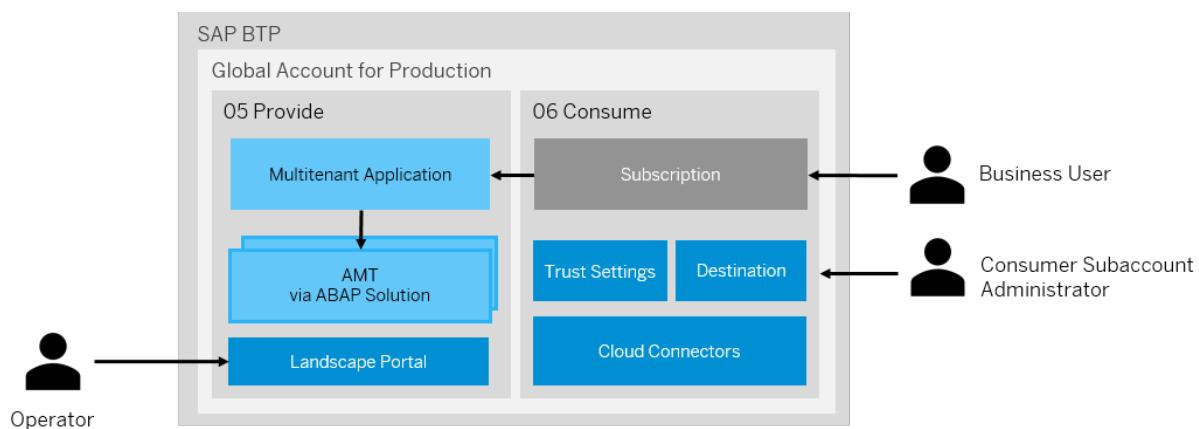
For development and maintenance processes, the steps mentioned below, that are similar to the ones described in [Use Case 2: One Development and Correction Codeline in a 5-System Landscape \[page 747\]](#), are performed.



- ABAP system COR and QAS have the same software state, unless a new change is tested and released. This means, transport requests are released in ABAP system DEV only if development is completed and it's planned to import the changes to the production ABAP system.
- Upon cutoff date, development is finished. All development that is released at this time must be tested and be of good quality. From then on, you must fix defects in the COR system and maintain them in parallel in the DEV system.
- Upon release date, all defects must be fixed. If, during testing in the QAS system, you make the decision that a complete functionality isn't delivered, developers must delete, revert, or disable the functionality in the COR system and release the corresponding transport requests. You can't remove objects from the release branch, e.g. by deselecting transport requests. To revert objects to a previous transported state, use the Compare editor of the Eclipse History view. If you want to withdraw the functionality in the DEV system as well, it's considered a correction and you have to perform double maintenance of corrections into the DEV system. See [Double Maintenance of Corrections into Development \[page 759\]](#).

- Users in ABAP system COR are locked during ongoing development and only unlocked when a correction has to be implemented
- For the consumption as a SaaS solution, instead of importing a release branch into a productive system, software components are installed via add-on delivery packages into multitenancy-enabled production systems AMT provisioned via the ABAP Solution service. See [ABAP Solution Service \[page 1379\]](#).

## Global Account for Production



### 1. 00 Landscape Portal

Since the Landscape Portal service is only available in region eu10, we recommend creating a dedicated subaccount in this region. This subaccount will contain your Landscape Portal & CI/CD service subscriptions for you to freely choose other regions for your other subaccounts.

#### ⓘ Note

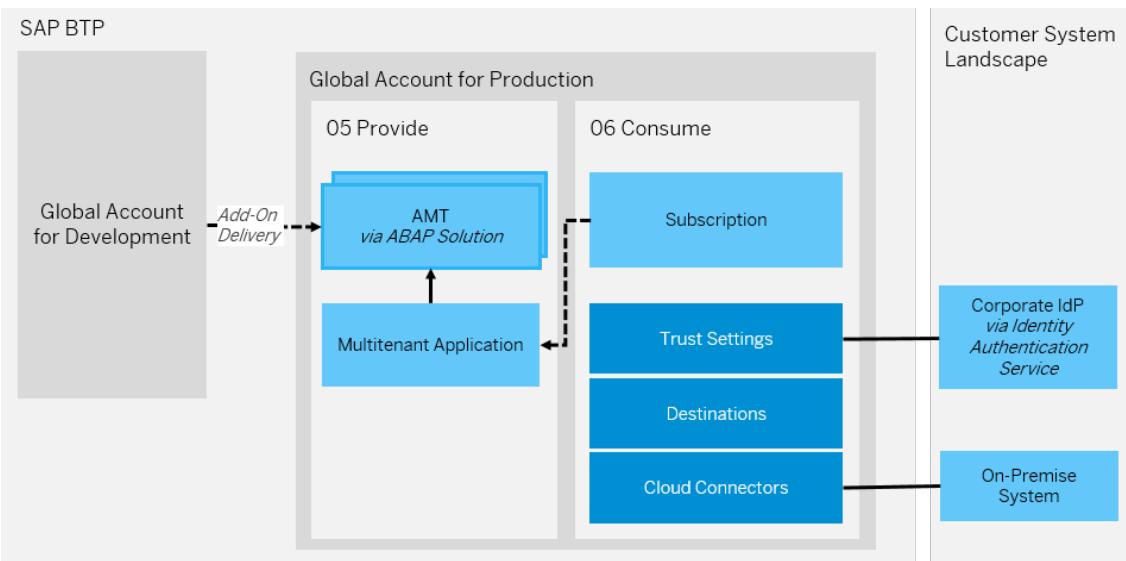
If you plan to create any of the other subaccounts in eu10, then you can simply reuse one of those subaccounts.

### 2. 05 Provide: Provider subaccount including provider space

The multitenant application is deployed to the provider subaccount in the global account for production. The ABAP solution service can then provision abap / saas\_oem systems (AMT), tenants, and users in this account once a consumer subscribes to the provided SaaS solution.

### 3. 06 Consume: Consumer subaccount

For each customer, a consumer subaccount is created in the global account for production.



This allows consumers to have their own configuration of:

- Trust settings (custom identity provider).

#### ⓘ Note

If you want to integrate an existing corporate identity provider for authentication/authorization in subaccounts of the global account for production, see [Trust and Federation with Identity Providers \[page 2204\]](#). To restrict access based on certain criteria, such as the IP address, you need to use the Identity Authentication service. See [SAP Cloud Identity Services - Identity Authentication](#).

- Connectivity via SAP Cloud Connector. See [Connectivity in the Cloud Foundry Environment](#).
- Destinations. See [Managing Destinations](#).
- Subscriptions

#### ⚠ Restriction

As of now, you can only expose SaaS applications for subscription from consumer subaccounts in the provider global accounts.

## 4.2.10.1.4.2 ABAP Environment Pipeline

The goal of the ABAP environment pipeline is to enable continuous integration for the ABAP environment.

The pipeline contains several stages and currently supports two relevant scenarios:

- Add-on build using AAKaaS
- Continuous testing via gCTS

A pipeline definition, containing all configuration, and a running CI/CD server are required to execute the ABAP environment pipeline. You can consume the different scenarios by enabling different pipeline stages in the configuration. For more information on how to configure the ABAP environment steps and stages, see [Configuration](#) and [ABAP Environment Pipeline](#).

The Landscape Portal offers the option to configure and execute the add-on build scenario using the [Build Product Version app](#), without the need for external repositories or Jenkins servers.

### ⓘ Note

The [Continuous Testing](#) scenario is currently not supported by the Landscape Portal. This means that a running Jenkins server is required, even when using the Build Product Version app. See [Add-On or gCTS](#).

## Jenkins Setup

### ⓘ Note

This is only required for running the Continuous Testing scenario. The add-on build scenario can be realized using the Build Product Version app in the Landscape Portal.

The general configuration of the ABAP environment pipeline always follows the same approach regardless of the scenario that you want to configure:

### 1. Prepare Git Repository

As a DevOps engineer, you need to prepare a Git repository by including the Jenkins file, initializing the pipeline, and the pipeline configuration file `.pipeline/config.yml`. See [Jenkins File](#) and [Technical Pipeline Configuration](#).

### 2. Create Service User for Git Repository

To enable read access to the Git repository, you need to create a service user and assign it to the repository. Later, this user's access credentials are stored in the Jenkins credentials by the Jenkins administrator. See [Using Credentials](#).

### 3. Create Jenkins Instance via Cx Server

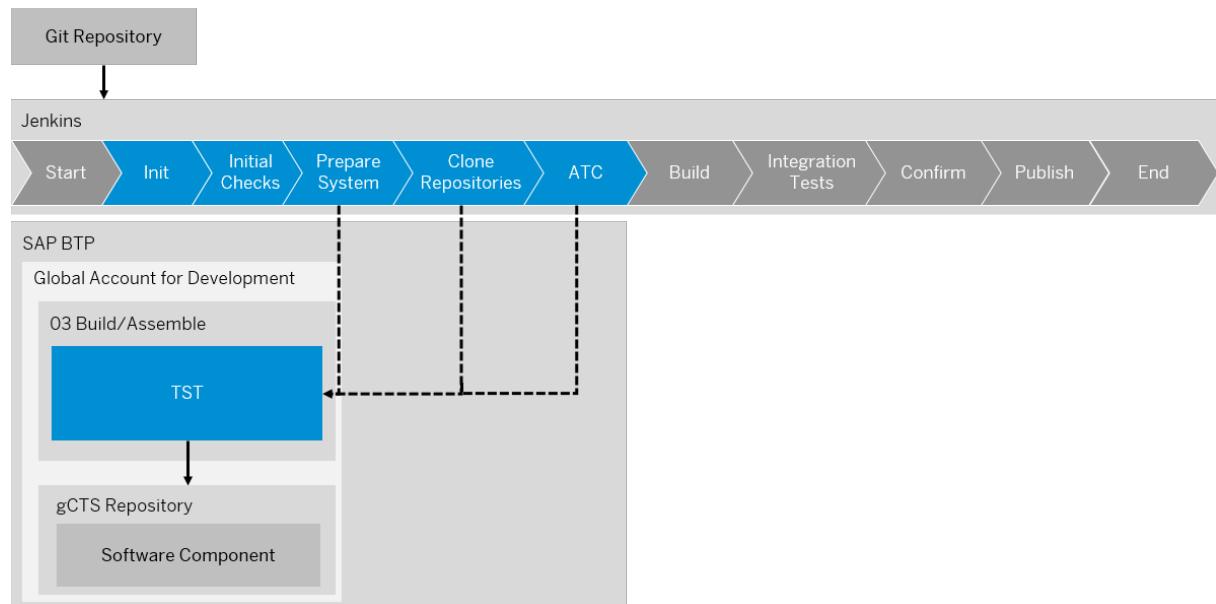
As a Jenkins administrator, you need to set up a new Jenkins instance using Cx Server Lifecycle Management. After initializing the Cx server that is based on a set of docker images, you can start the Jenkins server with `./cx-server start`. See [Cx Server](#) and [Docker Hub](#).

### 4. Configure Jenkins Instance

As a Jenkins administrator, you need to add technical Git user credentials and platform user credentials to the integrated secure store. Make sure that a shared library piper-lib-os is configured pointing to the project "Piper" library. See [Piper Library](#).

For more information on how to configure the ABAP environment steps and stages, see [Configuration](#).

## Transport from DEV to TST System via gCTS



Development system DEV and test system TST stay on the same main branch. To make the latest implementations available in the test system, the developed software components can be imported on a regular basis to test system TST and tested using the ABAP Test Cockpit by scheduling a planned execution of the ABAP environment pipeline. See [Continuous Testing on SAP BTP ABAP Environment](#) and [Running ATC Checks and ABAP Unit Tests on a Static ABAP Environment System](#).

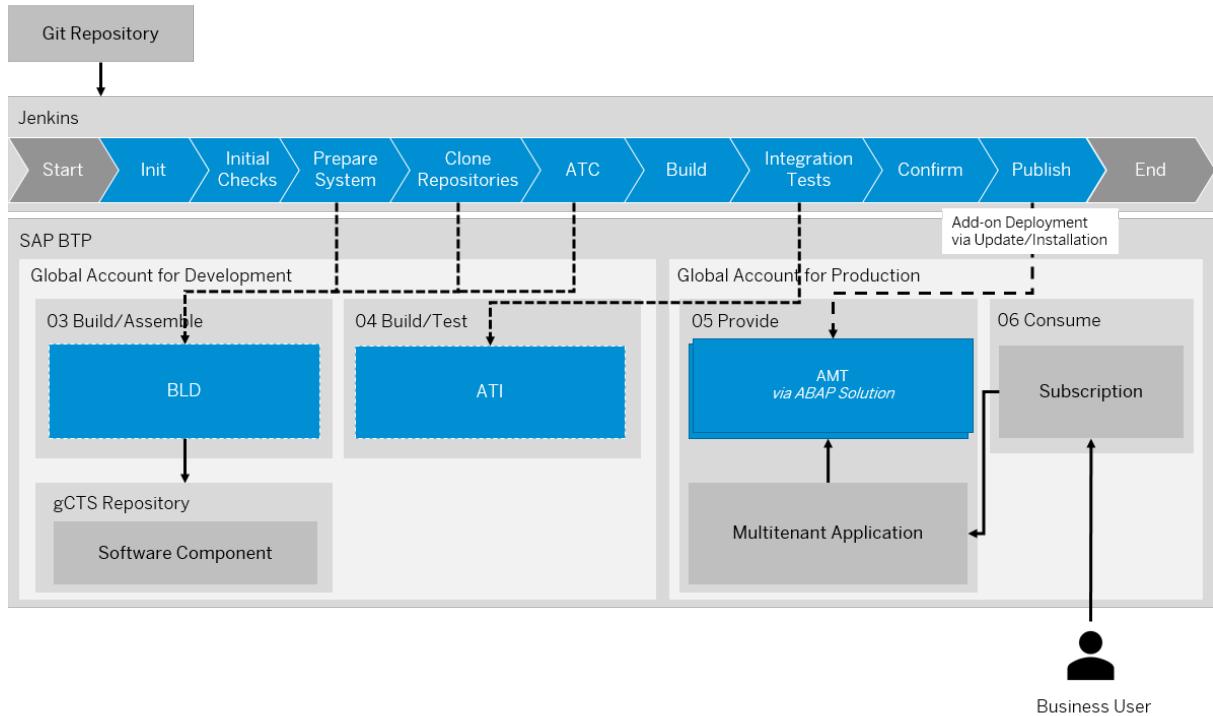
The ABAP environment pipeline is executed in a Jenkins server that is connected to subaccount **02 Test** in the global account for development by authenticating via a technical Cloud Foundry platform user. The permanent test system TST is used to import software components and run ABAP Test Cockpit checks.

## Build Add-On Version

The ABAP environment pipeline can automate the build and assembly process of ABAP add-ons. From creating the delivery packages in the assembly system to publishing the add-on release, all steps are part of this pipeline.

This scenario can be run in the Landscape Portal using the [Build Product Version app](#), or it can be configured manually and run on a local Jenkins server. In both cases, the same pipeline stages will be executed.

See [Build and Publish Add-on Products on SAP BTP ABAP Environment](#).



- <https://www.project-piper.io/pipelines/abapEnvironment/stages/initialChecks/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/initialChecks/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/prepareSystem/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/prepareSystem/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/cloneRepositories/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/cloneRepositories/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/ATC/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/ATC/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/build/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/build/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/integrationTest/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/integrationTest/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/confirm/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/confirm/]
- <https://www.project-piper.io/pipelines/abapEnvironment/stages/publish/> [https://www.project-piper.io/pipelines/abapEnvironment/stages/publish/]

The ABAP environment pipeline is executed in a Jenkins server that is connected to subaccount **03 Build/Assemble** and **04 Build/Test** in the global account for development by authenticating via a technical Cloud Foundry platform user. Transient systems are then provisioned for add-on assembly (BLD) and installation tests (ATI).

### Add-On Assembly System

Add-on assembly system BLD includes an instance of communication scenario SAP\_COM\_0510 and an instance of communication scenario SAP\_COM\_0582. See ([Deprecated](#)) [Test Integration \(SAP\\_COM\\_0510\)](#) [page 785] and [Software Assembly Integration \(SAP\\_COM\\_0582\)](#) [page 795]. Service keys with parameters referring to the communication scenarios are created in the BLD system, which leads to the creation of communication arrangements that can be used by the pipeline for inbound communication.

In the **Prepare System** pipeline stage, a new transient system BLD is provisioned for the add-on assembly. See [Prepare System](#) . ABAP Test Cockpit checks, software component imports, and the local build of deliveries are performed in this stage. By default, the system is deleted again in the **Post** pipeline stage. See [Post](#) .

The communication with remote components is enabled via communication scenario [Test Integration \(SAP COM 0510\)](#) and communication scenario [Software Assembly Integration \(SAP COM 0582\)](#).

By default, the system is created from scratch for each new add-on version. If you are configuring your pipeline manually, you can choose to reuse a permanent BLD system instead of provisioning a new one each time. This has the advantage of reducing the pipeline execution time, as system provisioning is one of the most time-consuming tasks. See [Build Add-Ons on a Permanent ABAP Environment System](#) .

To reduce costs, it is recommended that you shut down the permanent system when no builds are scheduled. See [Manage System Hibernation](#). Please note that hibernated systems still incur costs.

### Add-on Installation Test System

To verify that the delivery packages included in the add-on product version are installable, a target vector is published in "test" scope during the **Build** stage.

In the **Integration Tests** pipeline stage, a new transient system ATI is then provisioned for an add-on deployment test that installs the add-on product version. See [Integration Tests](#) . After the successful add-on installation is confirmed, the system is deleted, unless otherwise specified.

The communication with remote components is enabled via communication scenario [Test Integration \(SAP COM 0510\)](#).

By default, the ATI system is created from scratch for each new add-on version. If you are configuring your pipeline manually, you can choose to reuse a permanent ATI system instead of provisioning a new one each time. This has the advantage of reducing the pipeline execution time, as system provisioning is one of the most time-consuming tasks. We recommend that you provision a new ATI system at least for each new major release of your add-on.

### Add-on Assembly Kit as a Service

The Add-on Assembly Kit as a Service (AAKaaS) is used for registering and publishing the software product.

The service is offered in the SAP Service and Support systems, which means that access is granted via a technical communication user that packs the delivery into an importable package format. It is similar to the Software Delivery Assembler (SDA, transaction SSDA) as a part of the SAP Add-On Assembly Kit. See [SAP Add-On Assembly Kit](#) and [Add-On Assembly Kit as a Service](#) .

### Add-on Consumption

For the consumption of the add-on as a SaaS solution, software components are installed via add-on delivery packages into multitenancy-enabled production systems (AMT) provisioned via the ABAP Solution service. This is orchestrated via a multi-tenant application deployed to the Cloud Foundry environment on SAP BTP. See [Multitenant Application](#) and [Deploy](#).

#### ⓘ Note

If you need support or experience issues during the add-on build, see [Troubleshooting](#).

### 4.2.10.1.4.3 Multitenancy

The ABAP environment provides capabilities for multitenancy, meaning to host different customers on a single ABAP system. See [Multitenancy in the ABAP Environment \[page 1407\]](#).

#### → Tip

You should align your decision whether to enable multitenancy mode for your solution based on whether the [Multitenancy Development Guideline \[page 1412\]](#) can be followed during ABAP development.

### 4.2.10.1.4.4 Versioning and Branches

#### 4.2.10.1.4.4.1 Add-On Versioning

For the add-on delivery process, certain versioning principles apply. Learn how software components are combined to an add-on product and what versioning rules are used for both parts. Software components and add-on products follow a similar versioning pattern.

##### Add-on Product Version

Instead of importing software components into productive systems, when providing a SaaS solution in the ABAP environment, software components are packaged and installed into ABAP systems based on so-called add-on products/software products.

You can only install one add-on product for each system using the ABAP environment `saas_oem` service plan instead of the standard service. During provisioning of this service, parameters `addon_product_name` and `addon_product_version` are used to install a certain add-on product version into the system, after the system has been provisioned. See [ABAP Solution Service \[page 1379\]](#).

The add-on product name includes a reserved development namespace to separate the name from other customers/partners. See SAP note [84282](#).

The versioning pattern `<Release>. <Support Package Stack Level>. <Patch Level>` applies to the add-on product version.

#### ⓘ Note

Only in a new release, you can change the bundled software components of the add-on product.

Technically, the allowed number range is 1.0.0 to 9999.9999.9999.

For the release, there are no technical restrictions regarding the number of new versions. You only have to consider that the version has to be ascending across builds and there must be no gaps between versions. The initial release is always 1.0.0.

The support package stack level count starts again from 0 with each release.

The patch level count starts again from 0 with each support package.

While providing a specific add-on version, leading zeroes should be omitted (1.2.3 instead of 0001.0002.0003).

The add-on product version should be increased analogous to the version of the leading software component. The leading software component is the software component that is, as opposed to a reuse software component, exclusively used as part of one add-on product.

An exception to this recommendation is the patch level in the add-on production version: In case of an add-on product with a reuse software component, the patch level of the add-on product version might be higher than the patch level of the leading software component version.

For more information about software product versioning, see [Add-On Product Version](#).

## Software Component Version

Software components are self-contained and include packages as well as development objects. They are used in combination with a reserved development namespace to separate the name from other customers/partners. See [Software Components \[page 717\]](#).

The versioning pattern <Release>. <Support Package Level>. <Patch Level> also applies to the software component version.

### ⓘ Note

Software component versions are delivered with delivery packages. However, software component versions are not individual shipment entities. They can only be delivered to customers as part of a software product version.

Software component versions are only built once, independent from add-on product versions where the software component version was referenced. If an add-on product version is referring to a software component version that has already been created as part of a different add-on build, the created delivery package is reused.

Technically, the allowed number range is 1.0.0 to 9999.9999.9999.

For the release, there are no technical restrictions regarding the number of new versions. You only have to consider that the release has to be ascending across builds and there must be no gaps between versions. The initial release is always 1.0.0.

The support package level can only go up until 369 because of technical limitations. The support package level count starts again from 0 with each new release.

For the patch level, there is a technical limit of 36<sup>3</sup>, limited to 9999. The patch level count starts again from 0 with each support package.

For more information on software component versioning, see [Software Component Version](#).

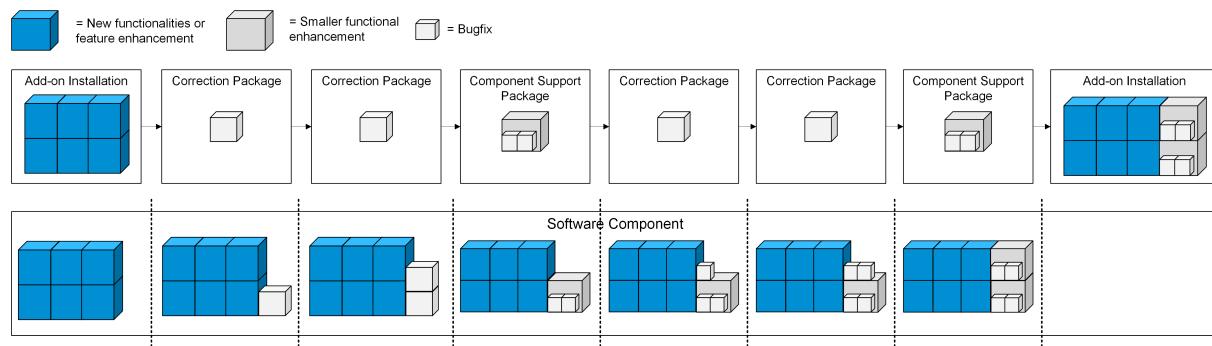
## 4.2.10.1.4.4.2 Add-On Package Types

For the delivery of add-on packages, there are three package types that serve different purposes: AOI, CSP, and CPK.

The different add-on package types are used for different purposes. The set of objects to be included depends on the package type.

### ⓘ Note

The required type of delivery is automatically determined by the delivery production tools based on how the version number of software components is changed.



### AOI (Add-On Installation)

Delivery packages of type Add-on Installation (AOI) are created for all new release versions, e.g. version 1.0.0, and should be used to deliver new functionalities or feature enhancements.

These delivery packages include all the objects in the software component and are usually created on a regular basis (e.g. quarterly).

### CSP (Component Support Package)

Delivery packages of type Component Support Package (CSP) are created for all new support package deliveries, e.g. version 1.1.0, and should be used to deliver a collection of patch deliveries or to deliver smaller functional enhancements.

These delivery packages include either the objects that were changed since the previous release delivery or since the previous support package delivery. They are usually created on a regular basis (e.g. bi-weekly).

### CPK (Correction Package)

Delivery packages of type Correction Package (CPK) are created for all new patch deliveries, e.g. version 1.0.1, and should be used to deliver bugfixes.

These delivery packages include only those objects that were changed since the previous patch delivery and are only created when necessary (e.g. emergency patch).

For more information on the add-on package types and how these are determined based on changes to the software component version, see [Software Component Version](#).

## 4.2.10.1.4.4.3 Branching

### ⓘ Note

Keep in mind that there are some restrictions for using branches in your development processes, for example the merging of branches is not supported and has to be done manually. See [ABAP Environment Specifics \[page 719\]](#).

Development in the ABAP environment is done in software components. For each software component, different branches can be created that include a different list of commits. For each system, only one branch can be active at a time. Branches are used to separate changes from each other that shall or might not be delivered together (development vs. correction).

To separate development of new features from the development of bug fixes for the active version of a software component, you can use branches for different purposes:

- **Main Branch:** Implementation of new planned features that are to be delivered in the future (development codeline)
- **Maintenance Branch:** Implementation of unplanned bug fixes for the active version of a software component that is deployed to customer systems (maintenance codeline)

The main branch is the default branch of each software component and therefore already exists when the software component is created. This branch is not deleted throughout delivery of different software component versions.

A maintenance branch should be created for each support package level of a software component and is named accordingly, for example v1.1.0 for support package level 1 in release 1. Maintenance branches can be deleted once the successor support package level is the currently deployed support package level in customer systems.

### → Recommendation

Create a new maintenance branch with each new support package level, and thus with each new AOI and CSP package delivered for a software component.

For the first release, the main branch of software components is used to create a maintenance branch for the add-on build process. In subsequent versions, the development and correction code line are separated by using the main branch and the maintenance branch. See [Use Case 2: One Development and Correction Codeline in a 5-System Landscape \[page 747\]](#).

All corrections to a software component based on a certain support package level are released in the corresponding maintenance branch and need to be maintained as new commits in the main branch (double maintenance). See [Double Maintenance of Corrections into Development \[page 759\]](#).

For each ABAP system, only one branch can be active at a time. Therefore, in development system DEV and test system TST, the main branch of a software component is checked out. Whereas in correction system COR and quality assurance system QAS, the currently relevant maintenance branch is checked out.

## 4.2.10.1.4.4.4 Delivery Models

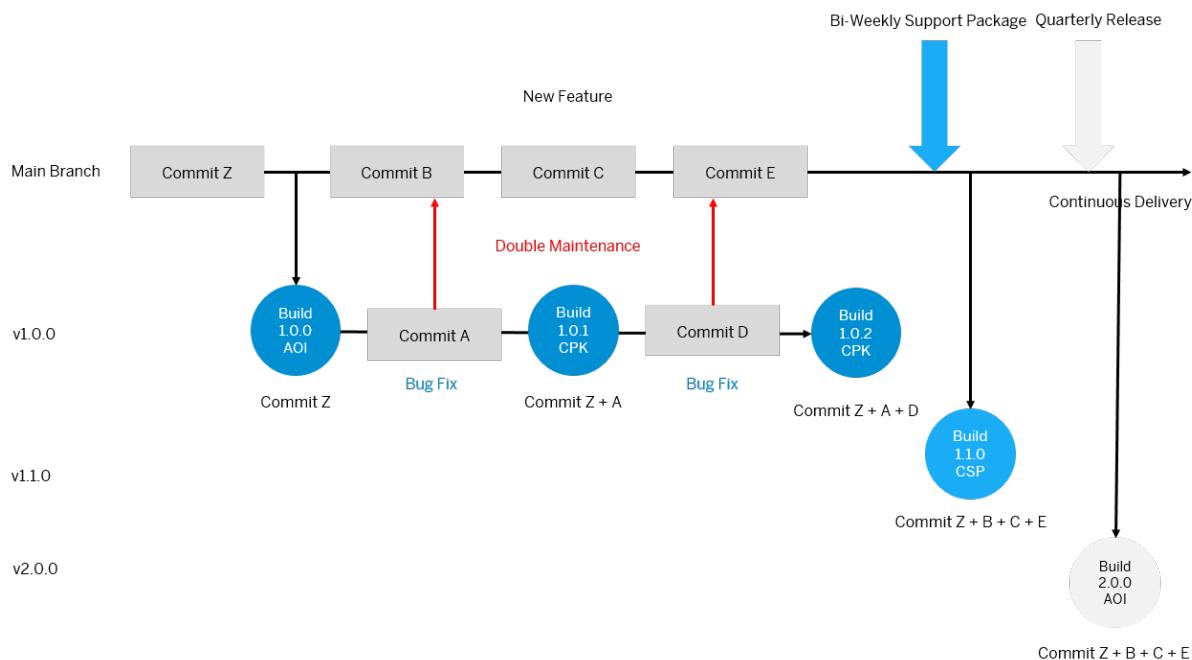
Based on the mentioned software component versioning principles, the add-on package types, and branching approach, you can use different delivery models.

### → Recommendation

You should choose the delivery model based on the existing processes in your development/shipment environment. Shorter delivery cycles have the advantage that new features are delivered more frequently. Therefore, we recommend following the continuous delivery approach whenever possible.

### Continuous Delivery

With the continuous delivery approach, new features should be delivered as soon as possible: In this case, new features can be shipped with support packages more often than new releases of a software component. These short development cycles ensure that changes can be deployed on short notice and in smaller increments. See [Continuous Integration and Delivery \(CI/CD\)](#).



In this example, commit Z represents all the commits prior to the first build.

The maintenance branch v1.0.0 is used for bug fixes that must be manually maintained in the main branch, which requires double maintenance to make sure that the development code line is up to date.

Commit Z and commit C are new features implemented in the development code line on the main branch. Development of these features is done in the DEV system, testing in the TST system.

Commit A and commit D are small corrections that are implemented in the correction code line on the maintenance branch for support package level zero of software component release one.

Development of these bug fixes is done in the COR system, testing in the QAS system.

After releasing these corrections in the maintenance branch, you have to maintain them as well in the main branch in the DEV system to secure consistency of changes in the correction and development code lines. As

a result, commit B and E are created in the main branch to reflect the corrections made in the maintenance branch (commit A and commit D).

The new support package build for version 1.1.0 is derived from the main branch and includes new features and bug fixes since the last support package level 1.0.0 has been released. With the new support package build for version 1.1.0 of the software component, a new maintenance branch v1.1.0 is created. All corrections for this support package level are developed on this branch.

For a new release, add-on version 2.0.0 is released. The resulting AOI package includes all features and bug fixes since the last AOI build for version 1.0.0 has been released.

Branch v1.1.0, that is based on the main branch, includes same commits as the main branch (Z, B, C, and E). For the build of CPK v1.0.2 and CSP v1.1.0, however, only the objects that have been changed since the previous CSP/AOI are included in the object list.

### **ⓘ Note**

With this delivery model, support packages not only include the corrections for the current support package level but also new features allowing shorter development cycles.

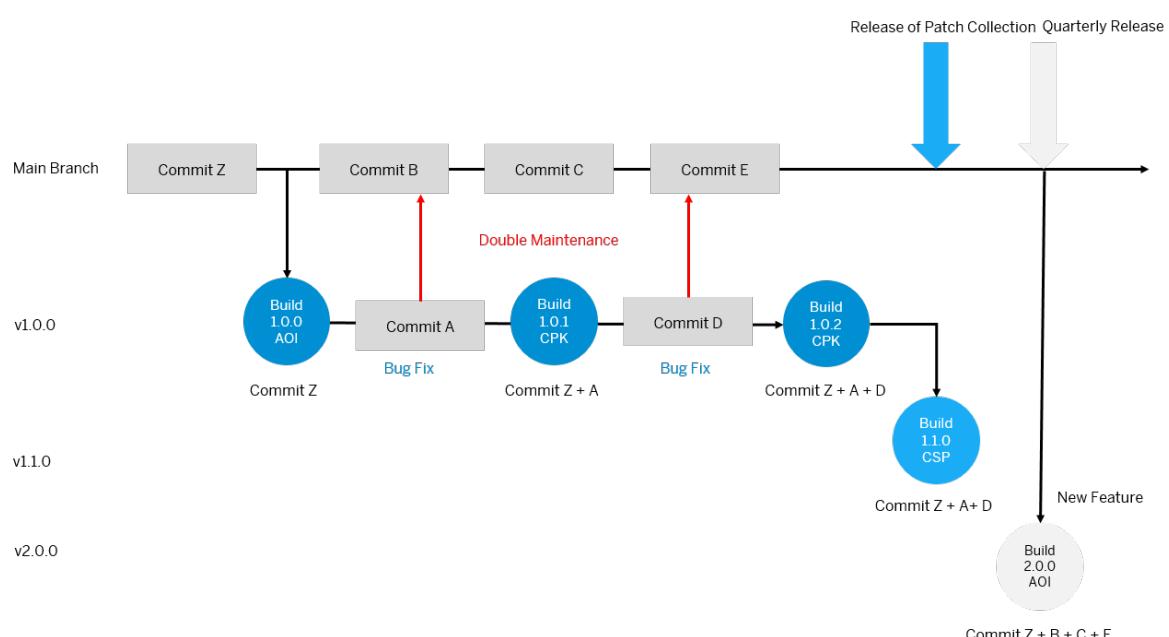
### **→ Recommendation**

In a continuous delivery model, we recommend:

- Building and deploying a new CSP on a weekly or biweekly basis.
- Shipping new software component releases on a quarterly basis. These AOI packages contain every object that has been built and delivered with the support packages, including all the patches.

### **Quarterly Shipment**

The quarterly delivery approach is recommended whenever longer test cycles are necessary and faster incremental development is not feasible. This might be required due to industry requirements, legal requirement etc.



Compared to the continuous delivery model, the maintenance branches for each support package level are not created based on the main branch but on the previous maintenance branch: e.g. branch v1.1.0 is based on v1.0.0.

That way, support packages are only used for the purpose of patch collections: only the corrections in commit A and D that were already delivered as patches in version 1.0.1 and 1.0.2 are included in support package 1.1.0.

#### ⓘ Note

With this delivery model, new features are only delivered with new release versions (AOI packages). The build for version 1.1.0 does not include commit C. As late as with the build for version 2.0.0, this feature is shipped. Consequently, new features take longer until they are available in production systems.

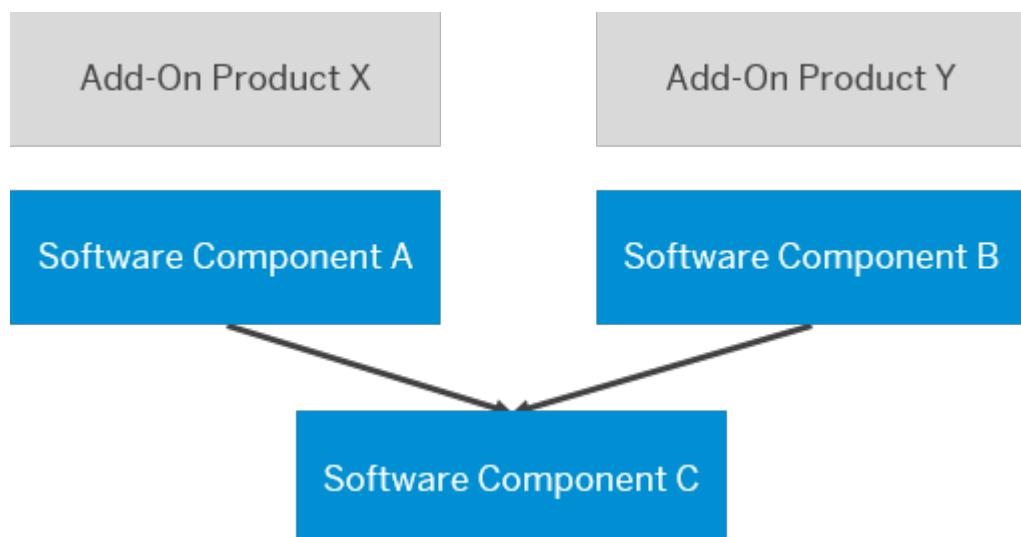
This delivery model is similar to processes commonly used to ship on-premise solutions where upgrades are much more complex and usually performed on a quarterly basis. Instead of shipping more often but in smaller packages, this approach promotes building larger shipment packages with a lot of new features.

Branch v1.1.0, that is based on branch v1.0.0, includes the same commits as branch v1.0.0 (Z, A, and D). For the build of CPK v1.0.2 and CSP v1.1.0, however, only the objects that have been changed since the previous CSP/AOI are included in the object list.

### 4.2.10.1.4.4.5 Reuse Software Components

Add-On products can consist of multiple software components. However, objects of one software component can't be used in another software component by default. Software components provide their functionality to other software components via explicitly released APIs. This means, you must set the API state of an object to [Released](#) if you want to use it from another software component. See [Released APIs](#) and [Finding Released APIs and Deprecated Objects](#).

Having this in mind, you can structure development into reuse components that are used across several components.



In this example, one add-on product consists of a leading software component A and software component C, whereas another add-on product consists of a leading software component B and C. Software component A and B use released objects of software component C, and thereby depend on this software component.

Once you have released an object, we recommend not changing it incompatibly because this can lead to errors for its consumers.

Software components must not have cyclic dependencies. In the example above, reuse software component C must not use objects from leading software component A. For more information on software component restrictions, see [Software Components \[page 717\]](#).

In the add-on descriptor file, dependencies are reflected by the order of the components in the repositories list:

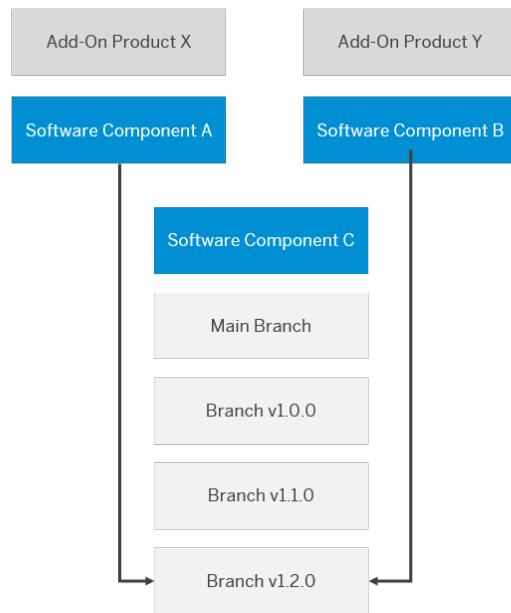
```
---  
addonProduct: "/NAMESPC/PRODUCT_X"  
addonVersion: "1.2.0"  
repositories:  
  - name: "/NAMESPC/COMPONENT_C"  
    branch: "v1.2.0"  
    version: "1.2.0"  
    commitID: "7d4516e9"  
  - name: "/NAMESPC/COMPONENT_A"  
    branch: "v2.0.0"  
    version: "2.0.0"  
    commitID: "9f102ffb"
```

In this case, component `/NAMESPC/COMPONENT_C` is the reuse component. It has no dependencies to any other top components and is therefore listed first. This is important for the right order during software component import. `/NAMESPC/COMPONENT_C` needs to be imported before `/NAMESPC/COMPONENT_A` to avoid import errors. In addition to that, software components in the bundle can use different namespaces in the software component name.

#### → Recommendation

In a software component bundle with a reuse component, this component is assembled whenever the software component version is defined for the first time in the add-on descriptor file. Therefore, make sure to define the correct software component version to prevent building the wrong software component version for `/NAMESPC/COMPONENT_C`.

We recommend following the same delivery model (identical shipment cycles) for all involved software components in a delivery scenario with a reuse component.



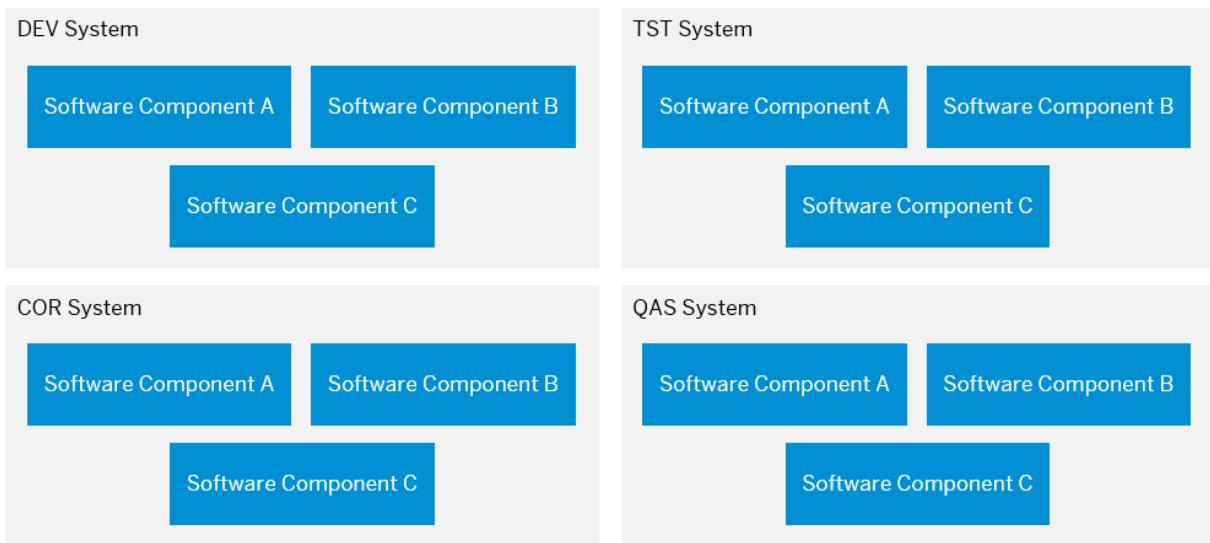
In this scenario, add-on product X and Y are based on the continuous delivery model and still undergo development. That means, for both add-on products, a new version can be shipped on a bi-weekly basis.

Consequently, the leading software components A and B refer to objects in the same branch of the reuse software component C, for example branch v1.2.0

Let's assume that development and correction of both software components is performed in one DEV and COR system. If there has to be a bug fix in reuse software component C, corrections to the corresponding software components A and B are still possible. This is due to the fact that branch v1.2.0 can remain checked out in software component C, while these changes are made.

The same applies to the development process: A new feature can be developed in leading software component A using the main branch of reuse software component C. Since only the main branch of reuse software component C is used in the DEV system, the development in the corresponding software components A and B can continue in parallel.

This scenario is resource-efficient: only one system is required for development in system DEV and one for corrections in COR. With respect to the developer experience, it is more convenient to have all implementation components in one development system because everything is in one place and the whole system landscape is easily understandable.



The system setup regarding the testing and quality assurance for corrections depends on how you want to test the add-on products:

- Do you want to provide a test environment that is identical to a production system with the add-on product installed? Then you should establish one TST and one QAS system for each add-on product in the sense that only those software components used for each add-on product are imported into these systems.
- Do you want to reduce the number of systems without restricting the test environment? Then you only need one TST and one QAS system, where all software components are imported.

#### → Recommendation

The number of required systems for DEV, COR, TST, and QAS depends on the number of branches that are actively used in dev, test, and build processes. Whenever more than one branch is actively used, one ABAP system is required per branch because an ABAP system always checks out only one branch of a software component.

### 4.2.10.1.4.5 Delivery via Add-On or gCTS

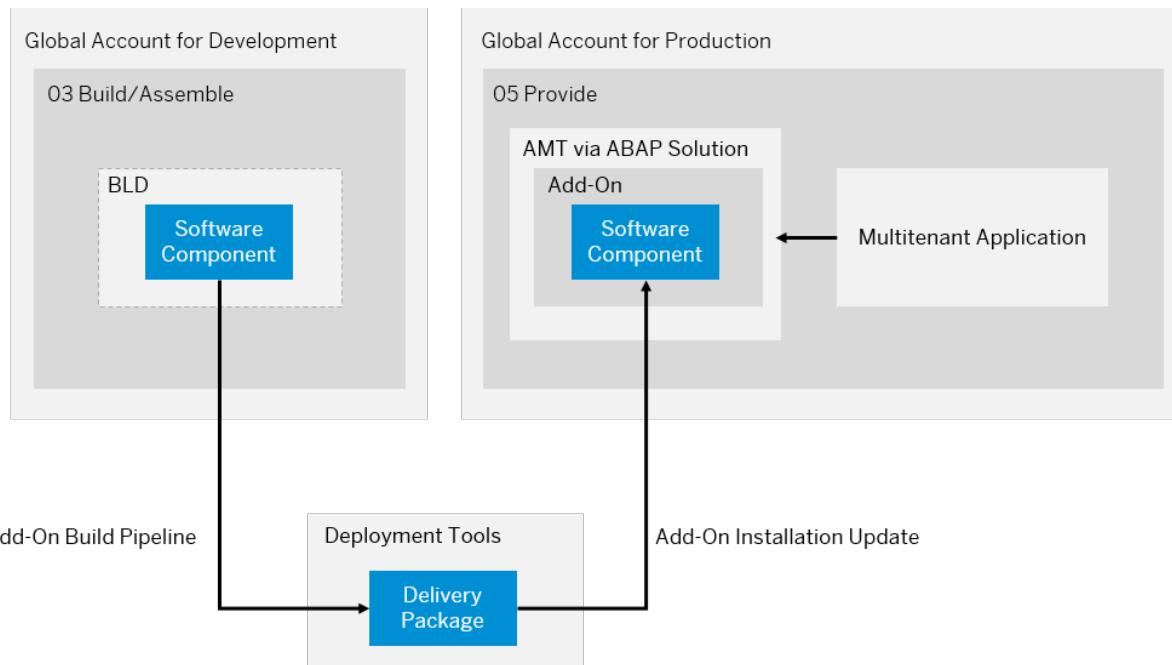
For SaaS solutions in the ABAP environment, software components including ABAP development objects have to be delivered to your production system. This is referred to as upstream process.

Afterwards, the SaaS solution is provided by offering a multitenant application that facilitates a subscription mechanism. This is referred to as downstream process.

There are multiple options to deliver software components to the systems. The delivery process described in this documentation mainly focuses on using add-on products as a means to deliver software components by installing or updating delivery packages in a system. See [The Add-On Product](#).

As an alternative, you can choose a delivery process using the gCTS-transport-based import of software components. See [Transport Mechanisms \[page 716\]](#).

## Add-On Build Delivery Process



An add-on is a bundle of software components. To create an add-on, a build process orchestrated by an add-on build pipeline has to be triggered.

With delivery tools, the software components are packaged into deliveries that are then further processed. Eventually, the deliveries are stored as delivery packages using the deployment tools provided by SAP.

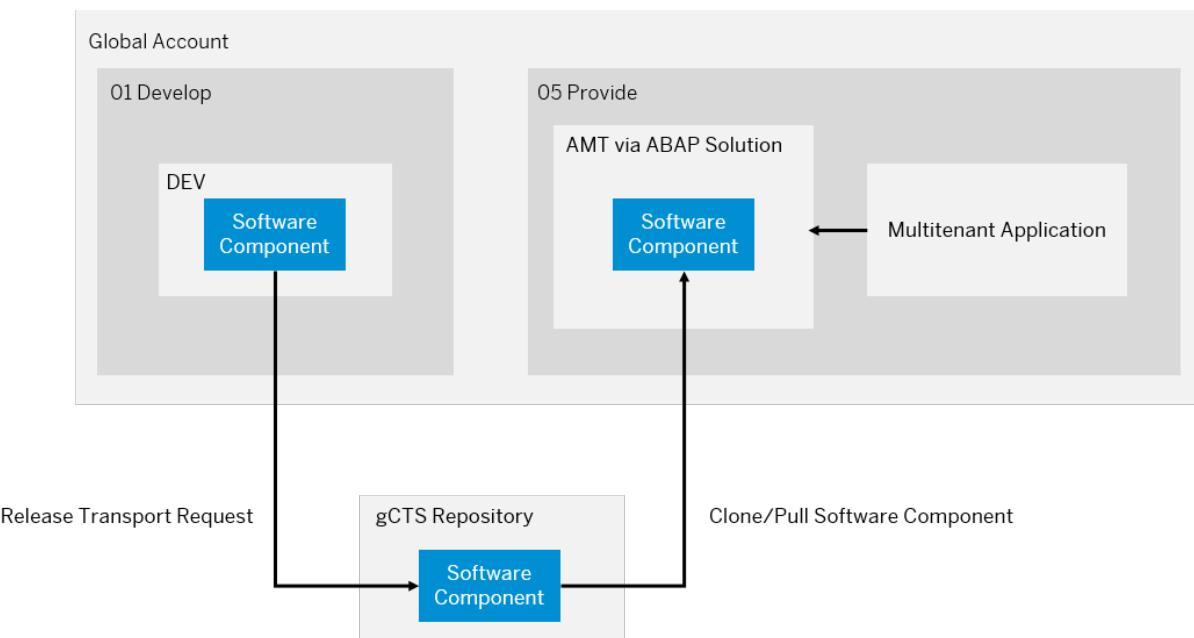
Add-on deployment is either performed by using an add-on installation during system provisioning or as an add-on update triggered in the *Landscape Portal* application.

Delivery packages are of different types, which means that the build and deployment of these packages differ accordingly. For example, the import of a software component for an add-on installation package or delta-import for correction packages.

BLD and AMT systems are based on different service plans: `abap/standard` for a plain ABAP system and `abap/saas_oem` for an ABAP system with installed add-on. The usage of these service plans is separated in two different global accounts for development and production purposes. Making software components available across different global accounts is only possible by using the add-on delivery process.

Making software components available across different global accounts is only possible by using the add-on delivery process. You can't use software components imported via add-on in the *Manage Software Components* app.

## gCTS Transport Delivery Process



The gCTS-transport-based delivery process takes place in one global development account only. This is due to the fact that the underlying gCTS repository is only shared among systems in the same region in the same global account.

ABAP development objects are released with a transport request in a particular software component in the development system. The transport request release automatically exports the changes into an SAP-managed gCTS repository by creating a new commit. Once the changes are exported into the gCTS repository, they are centrally available and can be imported into any other systems in the same region in the same global account. By cloning a software component into a customer production system using the [Manage Software Components](#) app, implemented ABAP objects become available to customers subscribed to the multitenant application.

The customer production system AMT is provisioned during the initial subscription to the multitenant application. Since the SaaS solution needs to include the implemented software, software components must be cloned right after the first subscription.

Add-On Build Delivery (Productive Build Pipeline)

gCTS Transport Delivery (Transport Mechanism)

<b>System Landscape/Account Model</b>	<ul style="list-style-type: none"> <li>ABAP systems for development purposes (development, test, demo) are of type <code>abap/standard</code> or <code>abap/abap_saas_oem</code>, and are provisioned in the global account for development based on a discounted development license</li> <li>ABAP systems for production purposes (used by partner customers) are of type <code>abap/standard</code> or <code>abap/abap_saas_oem</code>, and are provisioned in the global account for production based on a production license</li> <li>Software components are made available across these different global accounts by using add-ons</li> <li>For execution of the add-on build pipeline, an add-on assembly system BLD and add-on installation test system ATI are (temporarily) required</li> <li>All ABAP systems are of type <code>abap/standard</code> and are provisioned in the same global account</li> <li>No additional ABAP systems, apart from already existing development and test systems, are required to facilitate the delivery</li> <li>Single global account because gCTS repositories are only shared among ABAP systems in the same region in the same global account</li> </ul>
<b>Ease of Use</b>	<ul style="list-style-type: none"> <li>Initial efforts to set up ABAP environment pipeline for add-on build</li> <li>ABAP environment pipeline needs to be executed in a Jenkins CI/CD Server</li> <li>Initial configuration effort and requires to set up Jenkins server infrastructure. See <a href="#">Infrastructure</a>.</li> <li>Resulting delivery packages are either installed as part of an add-on during system provisioning or updated centrally in the systems using the <a href="#">Landscape Portal</a> application. See <a href="#">Perform Add-on Updates</a>.</li> <li>No build process required; deployment triggered locally using <a href="#">Manage Software Components</a> app</li> <li>Changes to software components are released and exported to a central remote repository using transport requests</li> <li>The <a href="#">Manage Software Components</a> app, that is part of every ABAP system, is used to trigger the import of new changes locally for each system</li> <li>As an alternative, software component import can be triggered by using the ABAP environment pipeline. See <a href="#">ABAP Environment Pipeline</a>.</li> </ul>

<b>Quality Measures/Resilience</b>	<p>Add-on build pipeline for delivery to production provides a guided process with strict checks:</p> <ul style="list-style-type: none"> <li>• ABAP Test Cockpit checks before transport release</li> <li>• Optional: ABAP Test Cockpit checks as part of ABAP environment pipeline for continuous testing</li> <li>• ABAP Test Cockpit checks as part of ABAP environment pipeline for add-on build</li> <li>• Checks during add-on build</li> <li>• Add-on installation test</li> <li>• Check for minimum platform version during add-on installation/update</li> <li>• Checks for import conditions during add-on installation/update that prevent import of unwanted software component states</li> </ul>	<p>gCTS transport includes basic checks, for strict checks additional processes need to be established:</p> <ul style="list-style-type: none"> <li>• ABAP Test Cockpit checks before transport release</li> <li>• Optional: ABAP Test Cockpit checks as part of ABAP environment pipeline for continuous testing</li> </ul>
<b>Auditability/Traceability</b>	<ul style="list-style-type: none"> <li>• Reliable, secure, traceable, and automated process based on add-on build using ABAP environment pipeline</li> <li>• Resulting delivery packages created as part of add-on build are signed and stored permanently so that they can't be altered or deleted after a release</li> <li>• Import conditions are assigned to delivery packages so that it's technically verified, during add-on update/installation, that all prerequisite package requirements are met</li> <li>• Delivery packages can be traced back to a specific commit ID of a software component, also dependencies between delivery packages can be translated into dependencies between different commit IDs</li> <li>• Build logs are archived as part of the add-on build pipeline execution</li> <li>• Anonymization of build results (<i>Created by</i> information)</li> <li>• Add-on build pipeline guarantees that only tested changes are delivered to consumers</li> </ul>	<ul style="list-style-type: none"> <li>• Logs of software component import are not archived, only locally available in ABAP environment system</li> <li>• Commit IDs in software component branches can be deleted by accident if not contained in any other existing branch</li> <li>• Import dependencies between software components need to be taken care of manually, a specific import order must be adhered to</li> <li>• Software component states are only identified by a specific commit ID instead of a readable version number</li> </ul>

- Scalability**
- Online deployment only possible with add-on-based delivery
  - With the *Landscape Portal* as a central tool for orchestration of software lifecycle activities, the add-on version can be updated centrally in multiple systems at the same time
  - Add-ons are automatically installed during system provisioning of ABAP systems and application content is immediately available in systems after subscription to the multitenant application
  - Application content is initially not available: Software components need to be cloned locally into the provisioned systems using the *Manage Software Components* app. See [How to Clone Software Components \[page 2808\]](#).
- ① Note**
- You can use a separate test subscription that is not related to a consumer to trigger the creation of the system before the consumer subscribes to the solution. This initial system creation and import of the software components needs to be performed for each consumer production system.
- Latest changes can be pulled locally into a system by using the *Manage Software Components* app. See [How to Pull Software Components \[page 2810\]](#).
  - Software component states installed in a system can only be identified locally in the *Manage Software Components* app.

SAP Software Certification	Possible, check details here: <ul style="list-style-type: none"> <li>• <a href="#">SAP Extension Suite Certification</a></li> <li>• <a href="https://www.sap.com/icc">https://www.sap.com/icc</a></li> </ul>	Not possible
Certifications and Compliance of Solution	SAP Compliance Offerings available in SAP Trust Center. See <a href="https://www.sap.com/about/trust-center/certification-compliance.html">https://www.sap.com/about/trust-center/certification-compliance.html</a> .	

<p>Requires less effort to be compliant because a strict process, supported by SAP delivery/deployment tools, is provided.</p>	<p>Requires additional effort to be compliant because a strict process needs to be established without SAP tooling support.</p>
--	---

#### → Recommendation

We recommend using the **add-on build delivery** process because it offers a bundling that can be used to implement a sophisticated delivery approach with separate codelines, versioning, and, as a result, different delivery package types. See [Delivery Models \[page 1337\]](#). It also comes with a separation of ABAP system licenses for development and production purposes, with the development systems being available at a discounted rate.

Example scenarios: Long-term projects with frequent maintenance deliveries, expected growth of consumers over time, and adherence to specific standards/certifications.

The **gCTS transport delivery** approach is a more lightweight approach that might be used especially if a simple setup and quick readiness are important. Ease-of-use of the approach comes with disadvantages in auditability, traceability, scalability, and resilience. These trade-offs might be resolved by establishing additional manual process steps or documentation requirements.

If the multitenant application is configured with `tenant_mode = single`, a dedicated system is created per subscription. See [ABAP Solution Service \[page 1379\]](#). In that case, gCTS delivery is only recommended if the SaaS solution is expecting a small set of customers or if the subscription is performed together with the customer. Otherwise, the effort for executing manual tasks, such as importing software components would be too high.

Example scenarios: Prototyping projects, quick time-to-market launch, in-house solutions, predictable number of consumers, and no expected exceptional growth rate

Once a first customer is using the SaaS solution and a first production system is provisioned, there is no migration path available to switch from one approach to the other, which means that systems would have to be deleted. However, it is likely to start out with gCTS transport delivery in the prototyping phase of a project and establish an add-on build delivery approach once the initial version needs to be delivered and customer systems need to be created.

### 4.2.10.1.5 Develop, Test, Build

To develop and provide add-ons, you have to set up a suitable system landscape and Cloud Foundry environment account structure.

You need a partner contract to create the account structure. The accounts for the upstream (add-on development, test, build) and downstream (providing add-on as SaaS solution) parts of the process need to be separated from each other and therefore are handled in dedicated global accounts. We refer to these as global accounts for development and production account. For more information regarding the development license, see [SAP PartnerEdge Test, Demo & Development Price List](#). To find out more about production licenses, see [SAP PartnerEdge: Resources for OEM Partners](#).

→ Tip

For in-depth information about the system landscape/account model, check out [System Landscape/Account Model \[page 1323\]](#).

For the initial add-on delivery, you should create a development and test landscape following the guidelines mentioned in [Concepts \[page 1323\]](#). Once backend and frontend development is completed, you can test the add-on implementation in a persistent test system. Upon successful completion of this step, set up a CI/CD server along with a Jenkins pipeline to execute the add-on build allowing for an automated and efficient assembly. This automated pipeline can then be reused transforming all upgrade and update activities into a straightforward process.

Once the add-on build/assembly is finished, you can release the add-on for installation/update. First, the add-on build is installed in a test system in the global account for development to verify that it works correctly. If successful, you can provide the add-on to customers.

ⓘ Note

Besides using add-ons for delivering software components to production systems, gCTS transports can be used as an alternative approach. It is used for transporting software components between different ABAP systems.

See [Delivery via Add-On or gCTS \[page 1342\]](#).

#### 4.2.10.1.5.1 Access to Landscape Portal

The [Landscape Portal](#) acts as a central tool that allows you to perform lifecycle management operations, such as provisioning new consumers tenants, users, and more. See [Landscape Portal](#).

As an operator, subscribe to the [Landscape Portal](#) application in the [05 Provide](#) subaccount in both the global account for development and global account for production. The `LandscapePortalAdminRoleCollection` needs to be assigned to the users in the default identity provider of the subaccount that you want to provide access to the [Landscape Portal](#) app. See [Accessing the Landscape Portal](#).

ⓘ Note

Note: In the global account for development, you can use a booster (see [Booster for Landscape Portal \[page 1358\]](#)) to automatically set up a subaccount with a working Landscape Portal subscription.

ⓘ Note

In the Landscape Portal, you can access the provider system and choose the provider tenant (client 100). Via the provider tenant, you get access to Fiori Apps, for example the [Manage Software Components](#) app, which is used for importing software components via gCTS.

See [Delivery via Add-On or gCTS \[page 1342\]](#).

## 4.2.10.1.5.2 Prepare

Before you can proceed with the development of the add-on, you have to perform several preliminary steps.

- Register the development namespaces before any system is set up. To start the actual add-on development process, you have to set up the system landscape and global account structure for development. For pipeline automation, set up a CI/CD server, in this case a Jenkins server.
- Create one or more software components that make up the add-on in the namespaces registered for you. In this scenario, namespaces are required for both the add-on product name and software components. You can register these namespaces with the development namespace/license keys tool.
- Purchase the SAP BTP, Cloud Foundry environment entitlements necessary for the account setup, which includes the service ABAP environment for development. The ABAP environment (service plan `saaS_oem`), Application runtime, SaaS provisioning, ABAP Solution service as well as the Landscape Portal are used for providing the SaaS app.

### → Recommendation

We recommend using the browser-based IDE SAP Business Application Studio for UI development. In this case, additional configuration is required to enable developers to use this service.

Once you've completed these steps, you can start developing an add-on.

## Prerequisites

- To register a namespace, you need an S-user in SAP One Support Launchpad with authorization to reserve a development namespace. See SAP note [1271482](#).
- To set up global accounts for development and production, you need two global accounts on the SAP Business Technology Platform with the corresponding entitlements for services and applications. See [Entitlements and Quotas](#).
- To create ABAP instances, you have to set up the account structure in the global account for development, and assign entitlements for each subaccount. See [Configure Entitlements and Quotas for Subaccounts](#).
- To develop UIs, you need an entitlement for SAP Business Application Studio and an assignment to a subaccount for development. See [SAP Business Application Studio](#).
- To set up transports from the development to the test system, you need a test system and optionally a pipeline in a Jenkins CI/CD server that is provisioned using a Cx Server to automatically import new changes. See [Jenkins](#) and [Cx Server](#).
- To set up add-on development, you need a development system.

## 4.2.10.1.5.2.1 Register a Namespace

Using a reserved namespace for add-on development and build is necessary for a unique add-on product and software component name.

### ⓘ Note

You need to register a namespace before the first ABAP system is provisioned. To activate namespaces in already existing systems after system provisioning, see [Maintain Namespaces](#).

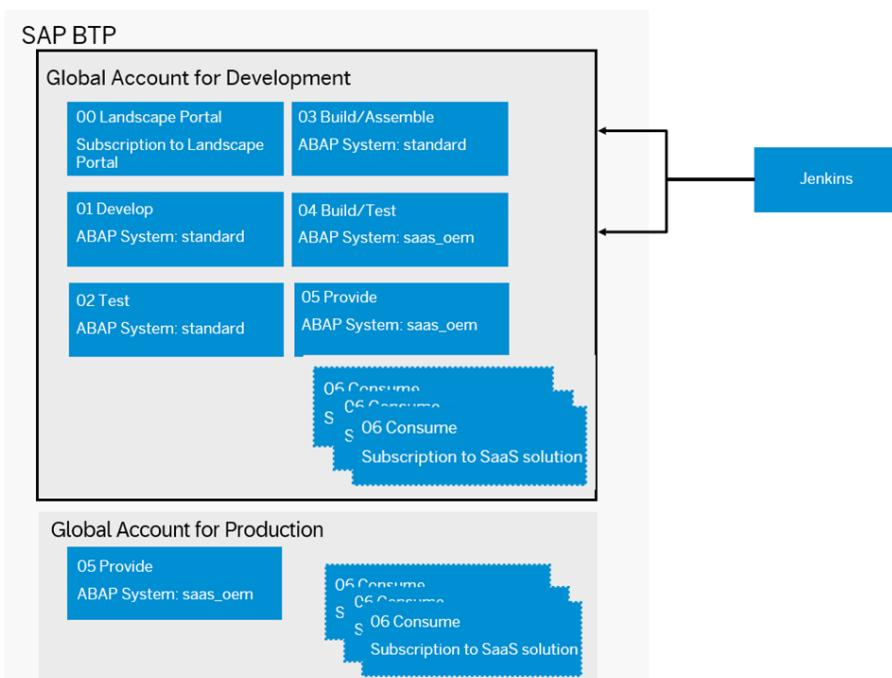
If you need a new S-user, get in touch with a user administrator in SAP ONE Support Launchpad.

As an S-user with authorization [Reserve Namespaces](#), you have to reserve a namespace for your partner customer ID to register a namespace with the [Namespace Application](#).

Due to length restrictions of some objects, namespaces should have 5–8 characters. See SAP note [105132](#) and [395083](#).

## 4.2.10.1.5.2.2 Set Up a Global Account for Development

As a SaaS solution operator, you have to configure the global account for development (used for development, testing, and demo purposes).



4

### → Recommendation

We recommend the following subaccount structure:

- In the [00 Landscape Portal](#) subaccount, your subscription to the Landscape Portal is created. If required, the CI/CD service should also be subscribed in this subaccount. No Cloud Foundry environment is required.
- In the [01 Develop](#) subaccount, add-on development is performed in a permanent development system. See [Development in the ABAP Environment \[page 705\]](#).

- In the *02 Test* subaccount, the developed software components are tested after a successful import into a permanent test system.
- In the *03 Build/Assemble* subaccount, the add-on package assembly is performed in a transient assembly system that is created and deleted automatically by the build pipeline.
- In the *04 Build/Test* subaccount, the add-on product is installed and tested again.
- In the *05 Provide* subaccount, the add-on product is provided as a SaaS solution for testing purposes in the development phase.

### Note

You can use a booster (see [Booster for Landscape Portal \[page 1358\]](#)) to automatically perform the setup of subaccounts 00 Landscape Portal, 03 Build/Assemble and 04 Build/Test. This includes the assignment of entitlements, creation of subscriptions and trust setup as described below.

### Note

If you use gCTS instead of add-ons for delivering software components to production systems, the setup and usage of the following subaccounts is redundant:

- *03 Build/Assemble* (used for the add-on assembly)
- *04 Build/Test* (used for add-on installation test)

Additionally, considering the availability of software components only in the same global accounts, you have to create the production systems as well as development and test systems in the same global account (only one global account is used).

In the provider subaccount, the entitlement for an ABAP instance of service plan type `abap/standard` instead of `abap/saas_oem` needs to be available.

See [Delivery via Add-On or gCTS \[page 1342\]](#).

You should configure a Cloud Foundry space in each subaccount. Dividing the development, testing, and assembling activities into different subaccounts allows for maximum flexibility. For instance, you may want to use different identity providers or consume different connectivity services during testing and development.

The ABAP systems that you use for development, testing, and add-on assembly are of type `abap/standard` and made available via entitlements. ABAP systems for add-on installation are of type `abap/saas_oem`. These service entitlements must be assigned by an administrator to different subaccounts, according to the following structure:

Global Account	Subaccount	Space	Services
Global Account for Development	01 Develop	Develop	1x abap/standard
			abap/hana_compute_unit (standard: 2)
			abap/abap_compute_unit (standard: 1)

Global Account	Subaccount	Space	Services
	02 Test	Test	1x abap/standard  abap/hana_compute_unit (standard: 2)  abap/abap_compute_unit (standard: 1)
	03 Build/Assemble	Build/Assemble	1x abap/standard  abap/hana_compute_unit (standard: 2)  abap/abap_compute_unit (standard: 1)
	04 Build/Test	Build/Test	1x abap/saas_oem  abap/hana_compute_unit (standard: 2)  abap/abap_compute_unit (standard: 1)
	05 Provide	Provide	1x abap/saas_oem  abap/hana_compute_unit (standard: 2)  abap/abap_compute_unit (standard: 1)  Application Runtime  abap-solution  saas-registry  xsuaa

Additionally, the following entitlements for SaaS application subscriptions are required:

- SAP Business Application Studio for UI development. See [SAP Business Application Studio \[page 239\]](#).
- Web access for ABAP for access to systems during development phase. See [Subscribing to the Web Access for ABAP \[page 186\]](#).
- Landscape Portal to manage systems and tenants in the provider subaccount. See [Landscape Portal](#).

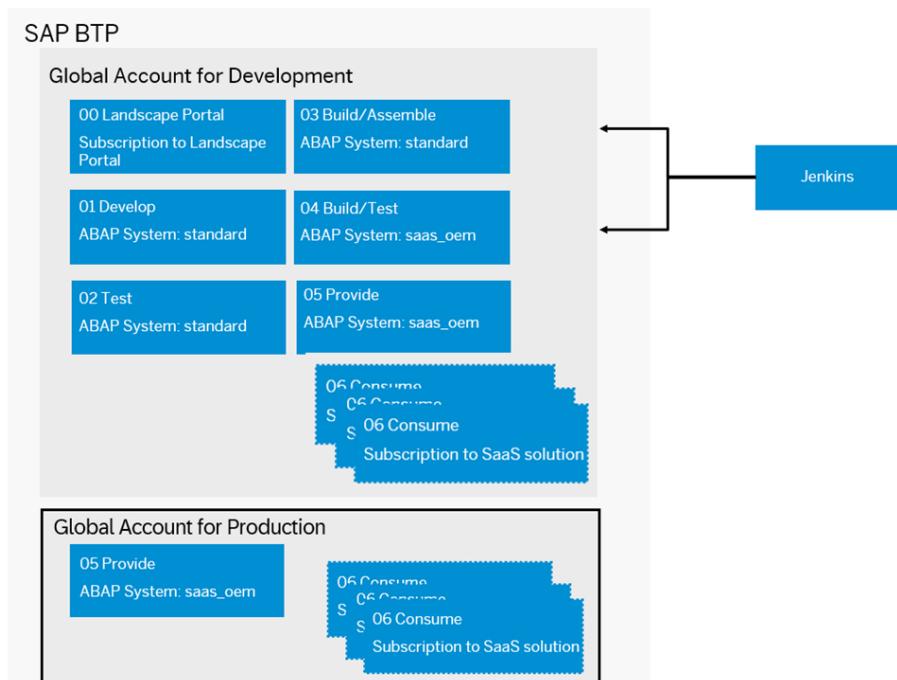
If you want to integrate an existing corporate identity provider in the subaccounts of the global account for development for authentication/authorization, see [Trust and Federation with Identity Providers \[page 2204\]](#). To restrict access based on certain criteria such as the IP address, you need to use the [SAP Cloud Identity Services - Identity Authentication](#).

→ Tip

For in-depth information about the system landscape/account model, check out [System Landscape/Account Model \[page 1323\]](#).

### 4.2.10.1.5.2.3 Set Up a Global Account for Production

As a SaaS solution operator, you have to configure the global account for production.



→ Recommendation

We recommend the following subaccount structure:

In the *00 Landscape Portal* subaccount, your subscription to the Landscape Portal is created. If required, the CI/CD service should also be subscribed in this subaccount. No Cloud Foundry environment is required.

In the *05 Provide* subaccount, the add-on product is provided as a SaaS solution for production purposes in the production phase. The solution is consumed by your customers from consumer subaccounts.

In the provider context, the ABAP environment (`saas_oem`) service plan is used.

ⓘ Note

In the provider subaccount, the entitlement for an ABAP instance of service plan type `abap/standard` needs to be available.

Additionally, considering the availability of software components only in the same global accounts, you have to create the production systems as well as development and test systems in the same global account.

See [Delivery via Add-On or gCTS \[page 1342\]](#).

These provider ABAP instances allow flexible sizing, multitenancy, and the possibility to install an add-on product during provisioning.

For the provisioning of these ABAP systems of service plan `saas_oem`, another set of services comes into play: With the ABAP Solution Provider and the saas-registry service, you can provide your add-ons as SaaS solution offerings. See [ABAP Solution Service \[page 1379\]](#).

Note that these service entitlements must be assigned to different subaccounts, according to the following structure:

Global Account	Subaccount	Space	Services
Global Account for Production	05 Provide	Provide	<code>abap/saas_oem</code>
			<code>abap/hana_compute_unit</code> (standard: 4)
			<code>abap/abap_compute_unit</code> (standard: 1)
			Application Runtime
			<code>abap-solution</code>
			<code>saas-registry</code>
			<code>xsuaa</code>

Additionally, the following entitlements for SaaS application subscriptions are required:

- Web access for ABAP for access to systems during the development phase. See [Subscribing to the Web Access for ABAP](#).
- Landscape Portal to manage systems and tenants in the provider subaccount. See [Landscape Portal](#).

If you want to integrate an existing corporate identity provider in the subaccounts of the global account for production for authentication/authorization, see [Trust and Federation with Identity Providers \[page 2204\]](#). To restrict access based on certain criteria such as the IP address, you need to use the [SAP Cloud Identity Services - Identity Authentication](#).

→ Tip

For in-depth information about the system landscape/account model, check out [System Landscape/Account Model \[page 1323\]](#).

#### 4.2.10.1.5.2.4 Create ABAP Instances

As a SaaS solution operator, you have to create ABAP instances.

For development and testing in the development codeline, one system is provisioned in each of the development and test subaccounts. Use service parameter `is_development_allowed` to differentiate between development and production systems. The test and assembly systems must be productive to avoid changes being made to the add-on outside of the development system.

Global Account	Subaccount	Space	ABAP Instances
Global Account for Development	01 Develop	Develop	Create an ABAP instance (abap/standard) DEV  Set parameter <code>is_development_allowed = true</code>
	02 Test	Test	Create an ABAP instance (abap/standard) TST  Set parameter <code>is_development_allowed = false</code>

#### ⓘ Note

You don't have to create an ABAP instance in the [03 Build/Assemble](#), [04 Build/Test](#), and [05 Provide](#) subaccount because the system is created automatically.

In the [03 Build/Assemble](#) account, assembly systems are provisioned by the add-on build pipeline. These systems are used to import the desired software components that are then packaged for add-on delivery (see [Software Assembly Integration \(SAP\\_COM\\_0582\) \[page 795\]](#)). As an operator, you don't have to provision these systems manually.

In the [04 Build/Test](#) subaccount of the global account for production, an add-on installation test system is automatically created. This ABAP environment service instance of plan `saas_oem` is provisioned with parameter `is_development_allowed = false`.

In the [05 Provide](#) subaccount of the global accounts for development and production, a customer production system is created automatically by the ABAP Solution Service during the first subscription. For more information on how to get started with your customer account, see [Getting Started with a Customer Account in the ABAP Environment \[page 166\]](#) and [Creating an ABAP System](#).

Use service parameter `is_development_allowed` to differentiate between development and production systems. The test and assembly systems must be productive to avoid changes being made to the add-on outside of the development system.

Subscribe to the Web Access for ABAP to gain access to the SAP Fiori launchpad in all subaccounts of the global production account.

## 4.2.10.1.5.2.5 Set Up UI Development

As a SaaS solution operator, you have to set up SAP Business Application Studio for development.

As a developer user, you can then create an SAP Fiori dev space and generate UI projects.

#### → Recommendation

For frontend development, we recommend using SAP Business Application Studio. See [Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#).

## 4.2.10.1.5.2.6 Set Up Add-On Development

Learn how to set up add-on development by creating and importing software components. Furthermore, read about configuring ABAP Test Cockpit checks and check variants, as well as enabling transport blocking to fix issues early on during development.

### Create and Import Software Components

To transport new developments from system to system, the add-on development is structured by software components. Software components are independent development containers.

To create a new software component for add-on development, as an add-on admin user, use the [Manage Software Components](#) app and create a new component of type Development. The name of the software component begins with a namespace that was activated in the development system. By default, all namespaces of the global account owner are enabled in the systems. See [How to Create Software Components](#).

Clone the main branch of the software component to the development and test system.

Software components in the development and test system should always stay on the main branch. If you want to work on maintenance branches to develop bug fixes and other maintenance deliveries, create a dedicated hotfix development and test system (maintenance codeline).

#### → Tip

For in-depth information about versioning and branches, check out [Versioning and Branches](#).

### Configure ABAP Test Cockpit Checks, Check Variants

By default, a check variant `ABAP_CLOUD_DEVELOPMENT_DEFAULT` is generated in ABAP environment systems. You should use this variant for ABAP Test Cockpit check runs or create a custom check variant based on the default check. See [Creating ATC Check Variants](#). You can also create custom ABAP Test Cockpit checks. See [Creating ATC Checks](#).

Custom ABAP Test Cockpit checks and check variants are created as part of a software component, whereas the default check variant is generated locally.

### Configure ABAP Test Cockpit to Interrupt Transport Releases

We recommend enabling the blocking of transport releases in case of priority 1 (error) findings using the default check variant `ABAP_CLOUD_DEVELOPMENT_DEFAULT` or a custom check variant. You can enable transport blocking in the ABAP Test Cockpit Configurator app. See [ABAP Test Cockpit Configurator](#).

Interrupting a transport release in case of severe findings can help to fix issues early on during development. The later errors are detected in the development process, the more costly it is to resolve them. See [Launching ATC Check Implicitly](#).

## 4.2.10.1.5.2.7 Set Up Transport from Development to Test System

You can perform the import of your software components into a test system either manually by using the [Manage Software Components](#) app or in an automated way using the [ABAP Environment Pipeline \[page 1328\]](#).

### Manual Import into Test System

As an add-on admin user, you can pull the latest released changes of a software component to the test system by using the main branch. You can test these changes in the test system independent from ongoing development. See [How to Pull Software Components \[page 2810\]](#) and [ABAP Lifecycle Management \[page 715\]](#).

### Automatic Import into Test System with ABAP Environment Pipeline

#### → Tip

For in-depth information about the ABAP environment pipeline used for automatic import into a test system, see [ABAP Environment Pipeline \[page 1328\]](#).



As a DevOps engineer, you can configure the ABAP environment pipeline for an automated import of software components to test system TST. With the pipeline, the pulling of specified software components is automated and performed on a regular schedule.

The continuous testing scenario of the ABAP environment pipeline is described in detail in [Continuous Testing on SAP BTP ABAP Environment](#).

## 4.2.10.1.5.2.8 Booster for Landscape Portal

One of the main use cases of the Landscape Portal is the building of product (add-on) versions. While the Landscape Portal hides much of the complexity involved in the process, it is required that a partner set up a suitable account model before starting.

The “Landscape Portal for SAP BTP ABAP Environment” booster is an automated process that automates the setup required to use the [Build Product Version](#) app in the Landscape Portal. Upon execution, it creates three new subaccounts corresponding to 00 Landscape Portal, 03 Build/Assemble and 04 Build/Test as described in [Set Up a Global Account for Development \[page 1351\]](#).

The three subaccounts fulfil the following roles:

- **00 Landscape Portal:** This subaccount will host your Landscape Portal subscription, which can be used for a wide range of lifecycle management tasks.
- **03 Build/Assemble:** This subaccount will host ABAP environment instances that are used to build new add-on product versions.
- **04 Build/Test:** This subaccount will host ABAP environment instances that are used to test the installation of new add-on product versions.

It is recommended to use a transient build system during the add-on build process. This means that with every build that is triggered, a new ABAP environment instance is provisioned and, after the process is finished,

it is deleted again. The booster also offers the possibility to provision such an instance during the booster execution, which can be later used during the build process.

You can start the booster from your global account for development. A wizard queries the user for the necessary data for the configuration of the subaccounts.

## Prerequisites

- Your global account for development should be assigned the necessary entitlements:
  - Landscape Portal (1x standard)
  - Continuous Integration & Delivery (1x standard)
  - abap (2x standard, 2x abap\_compute\_unit, 4x hana\_compute\_unit)
  - Web Access for ABAP (1x default)

## Procedure

1. Navigate to your global account for development in the BTP Cockpit.
2. Navigate to the *Boosters* tab.
3. Select the “Landscape Portal for SAP BTP ABAP Environment” booster and click *Start*.
4. Configure the *00 Landscape Portal* subaccount.
  1. Choose a subaccount name and subdomain.
5. Configure the *03 Build/Assemble* subaccount.
  1. Choose a subaccount name and subdomain.
  2. Choose a Cloud Foundry organization and space name.
  3. Specify a technical user from your identity provider that shall be used for the provisioning of build systems.
  4. Specify whether to provision a build system.
  5. If a system shall be provisioned, choose a system ID.
6. Configure the *04 Build/Test* subaccount.
  1. Choose a subaccount name and subdomain.
  2. Choose a Cloud Foundry organization and space name.
  3. Specify a technical user from your identity provider that shall be used for the provisioning of installation test systems.
7. Configure the user groups.
  1. Specify a list of administrator users. These users will be assigned authorizations as described in *Results* down below.
  2. Specify a list of viewer users. These users will be assigned authorizations as described in *Results* down below.

SAP ID Service is configured as the identity provider in all three subaccounts. If you wish to use different trust settings for different subaccounts, you can adjust the configuration as necessary after the booster execution.

### ⓘ Note

Currently, the Landscape Portal can only be used with SAP ID service as the application identity provider.

## Results

After successful execution, your global account for development should contain three new subaccounts with the following properties:

### 00 Landscape Portal

- Entitlements: Landscape Portal (1x standard), Continuous Integration and Delivery (1x default)
- Subscriptions: Landscape Portal, Continuous Integration and Delivery
- Administrator users have role collections *Subaccount Administrator* and *LandscapePortalAdminRoleCollection*.
- Viewer users have role collections *Subaccount Viewer*.

### 03 Build/Assemble

- Entitlements: abap (1x standard, 2x abap\_compute\_unit, 2x hana\_compute\_unit)
- Subscriptions: -
- Cloud Foundry is enabled
- 1 Cloud Foundry space is created
- If configured, an ABAP environment instance is provisioned
- Administrator users have role collection *Subaccount Administrator*, are *CF Org Managers* and *Space Managers*
- Viewer users have role collection *Subaccount Viewer*, are *CF Org Members* and *Space Developers*
- The specified technical user is granted the same authorizations as the viewer user group.

### 04 Build/Test

- Entitlements: abap (1x standard, 2x abap\_compute\_unit, 2x hana\_compute\_unit), Web Access for ABAP (1x default)
- Subscriptions: Web Access for ABAP
- Cloud Foundry is enabled
- 1 Cloud Foundry space is created
- Administrator users have role collection *Subaccount Administrator*, are *CF Org Managers* and *Space Managers*
- Viewer users have role collection *Subaccount Viewer*, are *CF Org Members* and *Space Developers*
- The specified technical user is granted the same authorizations as the viewer user group.

After executing the booster, you maintain the same technical user(s) in the *Credentials* section of the *Build Product Version* app.

You are then ready to configure one or more pipeline templates, see [Configure a Pipeline Template](#).

When configuring a template, reuse the resources created by the booster as follows:

For the "Prepare System" stage, reuse the subaccount/CF space from the **03 Build/Assemble** subaccount. If you provisioned a system during the booster execution, specify its instance name in the template so that it will be used.

For the "Integration Tests" stage, reuse the subaccount/CF space from the **04 Build/Test** subaccount. If you provisioned a system during the booster execution, specify its instance name in the template so that it will be used.

## 4.2.10.1.5.3 Develop

Both the initial implementation and any further development of the add-on are performed in the development system of the development subaccount.

You can use various SAP technologies at this point, such as the ABAP RESTful application programming model (RAP) for modeling and implementing business objects and the browser-based IDE SAP Business Application Studio for developing SAP Fiori UIs. Depending on the business use case, some additional development effort may be necessary to implement suitable launchpad content, Identity & Access Management artifacts, or communication management artifacts.

Once you've completed these development activities, the solution is ready to be tested in a suitable test system.

### Prerequisites

- For ABAP development, you need a developer user using ABAP development tools for Eclipse. See [Getting Started as a Developer in the ABAP Environment \[page 217\]](#).
- For UI development, you need a developer user using SAP Business Application Studio. See [Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#).
- For custom code migration, you need a business user that is assigned the business role based on business role template SAP\_BR\_IT\_PROJECT\_MANAGER, and a communication arrangement instance for SAP\_COM\_0464. See [Custom Code Migration \[page 2609\]](#).

## 4.2.10.1.5.3.1 ABAP Development

As a developer user, implement your custom business services with the ABAP RESTful application programming model. See [ABAP RESTful Application Programming Model](#). Maintain business catalogs (see [Identity and Access Management \(IAM\) \[page 797\]](#)) and communication scenarios (see [Communication Management \[page 883\]](#)) to expose services to business users and communication users. See [SAP Business Technology Platform \[page 7\]](#), [ABAP Environment Learning Journey](#) , and [ABAP Environment Community](#) . Also consider starting out with a free tier option for the ABAP environment to get hands-on development experience. See [Trial Accounts and Free Tier](#).

### Identity and Access Management

SAP Fiori applications and business services are represented by IAM apps and can be used to define the necessary authorizations. In an IAM business catalog, you bundle multiple IAM apps and their predefined authorizations, for example, for a specific business area.

Additionally, you can define business role templates to make it easier for administrators to find the relevant business catalogs to create business roles. See [Identity and Access Management \(IAM\) \[page 797\]](#).

## Communication Management

### ⓘ Note

Use customer-managed communication scenarios as a design time description to store technical information, such as inbound and outbound services and their service type, for example OData or SOAP by using the `create_by_comm_arrangement` method. See [Service Consumption via Communication Arrangements \[page 892\]](#).

Instead of coding against destination names, using communication scenarios allows consumers of the SaaS solution to create communication arrangements based on provided communication scenarios. Therefore, you don't have to create specific destinations in the consumer subaccount.

### ⚠ Restriction

The ABAP environment only supports one communication configuration layer intended to be used by the consumer. It's currently not possible to enforce a communication configuration exclusively managed by the service provider within the consumer tenant.

Services with the following protocols supported for **inbound communication** can be included as part of customer-managed communication scenarios:

- HTTP
- SOAP
- RFC Internet/Cloud Connector

### ⚠ Restriction

RFC is currently not supported for inbound communication in consumer tenants.

For **outbound communication**, the following protocols are supported:

- HTTP (Internet)
- HTTP (Cloud Connector)
- SOAP (Internet)
- SOAP (Cloud Connector)
- RFC (Internet)
- RFC (Cloud Connector)

See [Developing External Service Consumption](#).

## Business Configuration

Business configuration plays a major role in SaaS solutions. It refers to a predefined set of configuration options that affect its functionality and behavior. See [Business Configuration in SAP BTP ABAP Environment \(1\): Overview and BC Maintenance Apps](#).

To maintain these configuration options, you have to create dedicated apps using the ABAP RESTful Application Programming Model. See [Create a Business Configuration App for Factory Calendar Using the ABAP RESTful Application Programming Model](#).

Using the business configurations API, you can register business configurations. These business configurations are then displayed in the list of all maintainable business configurations in the SAP Fiori App

Maintain Business Configurations, if the user has the necessary authorizations for the service of the business configuration. See [Business Configuration Maintenance Object Cloud Platform API \[page 867\]](#).

### Key User Extensibility Enablement

Using SaaS solutions, you can provide key user extensibility for system-internal use (contract C1) and use in key user apps. This allows customers who use the solution to extend it to their specific requirements.

As a developer user, you have to implement key user extensibility in the development system in the Partner Development tenant (client 100) using ABAP Development Tools.

See [Providing Business Add-Ins \[page 1420\]](#) for guidance on how to prepare business add-ins (BAdIs) so that customers can add their own business logic in the solution by using the *Custom Logic* app.

See [Configuring Predefined Custom Fields \[page 2639\]](#) for guidance on how to equip your SaaS solution with support for customer-specific extension fields.

### Stability of Released APIs

Released APIs, such as BAdIs or predefined custom fields, must only be changed compatibly. Otherwise, for example, runtime errors might occur or an upgrade might fail. To ensure that released APIs are not changed incompatibly, run an ATC compatibility check. This check uses snapshots that are created using the [Manage API Snapshots](#) app and which are set to check-relevant, to check whether incompatible changes have been made to the released API. For more information, see [Manage API Snapshots \[page 2556\]](#).

### Multitenancy

With the ABAP environment, you can build multitenancy-enabled SaaS solutions. To do so, the add-on implementation has to follow certain guidelines. See [Multitenancy Development Guideline \[page 1412\]](#).

#### → Recommendation

If the add-on implementation is aligned with the development guideline for multitenancy, we recommend configuring your solution by setting `tenant_mode = multi` so that the same ABAP service instance is used for multiple consumers.

#### → Tip

For in-depth information about multitenancy, check out [Multitenancy \[page 1333\]](#).

## 4.2.10.1.5.3.2 UI Development

SAP Fiori applications are developed in SAP Business Application Studio on top of business services in the ABAP development system and then deployed to the ABAP development system to be part of the same software component as backend artifacts.

As a developer user, once business services are implemented as UI services, you can create SAP Fiori elements apps using SAP Business Application Studio. See [Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#).

#### ⓘ Note

To grant business users access to an app, the consumer's administrator creates corresponding business roles. Moreover, to influence the layout for specific user groups, the administrator can maintain

corresponding space and page templates. It's recommended to provide business role templates with business catalogs and corresponding space and page templates as part of an add-on. This simplifies the process and speeds up the configuration of the Fiori launchpad for each consumer.

As described in [Scoping Space and Page Templates \[page 1852\]](#), space and page templates need to be scoped manually in the development system. However, these are scoped automatically in consumer tenants.

### 4.2.10.1.5.3.3 (Optional) Code Migration from On-Premise

Optionally, you can migrate existing custom ABAP code for add-on development purposes. This custom code migration process analyzes your existing code for cloud-readiness. See [How to Check your Custom ABAP Code for SAP BTP ABAP Environment](#) and [Custom Code Migration \[page 2609\]](#).

After adapting the code and making necessary changes, you can migrate the code via abapGit. See [How to Bring your ABAP Custom Code to SAP BTP ABAP Environment](#).

### 4.2.10.1.5.4 Test

Once development has been completed, you can test the solution in the test subaccount that you have previously set up. The software components that make up the solution are imported into a provisioned test system, while any additionally required configuration has to be carried out by you. This may entail creating suitable business roles, maintaining communication arrangements, or other administrative tasks performed via the SAP Fiori launchpad of your ABAP system.

You can also use a CI/CD server and a Jenkins pipeline to automate the test process. This allows you to schedule regular tests and to be notified as soon as issues arise within the solution.

If the solution is successfully tested and works correctly, you can proceed with the add-on build.

## Prerequisites

- For testing in SAP Fiori launchpad in your ABAP environment, you need a business user in the test system that has the required authorizations to use the [Manage Software Components](#) app as well as authorizations that are required as a test user.
- For testing in the ABAP Test Cockpit, you need a developer user using ABAP Development Tools. See [Getting Started as a Developer in the ABAP Environment \[page 217\]](#).
- (Optional) For running ATC checks as part of the ABAP environment pipeline, you have to create a pipeline in a Jenkins CI/CD server that is provisioned using the Cx Server tool. See [Jenkins](#) and [Cx Server](#).

## 4.2.10.1.5.4.1 Test in the ABAP Environment SAP Fiori Launchpad

### Import Software Components

Before testing new developments in a software component in the test system, as an add-on admin, you have to import the latest changes from the remote repository. After you have cloned a software component into the test system TST, you can import the latest changes by pulling the software component in the [Manage Software Components](#) app. See [How to Pull Software Components \[page 2810\]](#).

### Create and Assign Business Roles

Before creating and assigning business roles, you have to make sure that business users are already available. They can be created manually or automatically. See [User Provisioning \[page 803\]](#).

To test the developed business services, as a test user, create business roles from the role templates in the test system and assign them to your user. See [Maintain Business Roles \[page 2666\]](#).

### Create Launchpad Space and Pages for Business Roles

To enable navigation to custom UIs via tiles, enable launchpad spaces and pages. Add spaces to the relevant business roles and add the needed SAP Fiori launchpad tiles to those spaces. Finally, enable spaces for business users in the system launchpad settings. See [How to Create Spaces and Pages for a Business Role \[page 2680\]](#).

### Create Communication Arrangements

To test inbound and outbound communication, create communication arrangements in the test system TST based on the implemented communication scenarios. See [How to Create a Communication Arrangement \[page 2596\]](#).

#### Note

Depending on whether you want to use an authentication method for outbound communication that requires a business user context (Oauth2 SAML Bearer Assertion, Oauth2 User Token Exchange, JWT Principal Propagation), you need to configure a destination in a communication arrangement instead of maintaining credentials by using an outbound communication user. See [Supported Protocols and Authentication Methods \[page 886\]](#) and [Create a Destination \[page 891\]](#).

You can integrate the test system TST with on-premise systems. See [Integrating On-Premise Systems \[page 1184\]](#).

In the subaccount of test system TST, the subaccount for testing, you can assign the Cloud Connector administrator role collection to the Cloud Connector administrator user to operate the data transmission tunnels used by the Cloud Connector.

### Create Business Configuration

As a test user, you can adjust business configuration objects in the [Maintain Business Configurations](#) app to change and influence the system behavior. See [Custom Business Configurations App \[page 2574\]](#).

### Configure Key User Extensibility

Key user extensibility that is enabled in the SaaS solution can be configured and consumed in test systems.

### ⓘ Note

Key user extensibility provided in a SaaS solution can only be configured in tenants of particular types, for testing purposes in tenants of type Partner Test.

These tenant types are provisioned in non-development systems, such as test system TST or quality assurance system QAS, where development is not allowed (`is_development_allowed = false`). The tenants are created independently from a subscription to the SaaS solution by using the [Landscape Portal](#) application. See [Manage Test Tenants](#).

As a test user in a Partner Test tenant (client >= 200), you configure key user extensibility in a test system.

- See [Custom Logic \(Deprecated\) \[page 2633\]](#) for guidance on how to use the [Custom Logic](#) app to create and maintain custom logic for business add-ins (BAdIs).
- See [Configuring Predefined Custom Fields \[page 2639\]](#) for guidance on how to configure predefined custom fields to customize applications and their UIs.

## 4.2.10.1.5.4.2 Test in the ABAP Test Cockpit

With the ABAP Test Cockpit, you can run a set of checks (check variants) on software component or package level. See [Checking Quality of ABAP Code with ATC](#) and [ABAP Test Cockpit in the Cloud – What is already possible](#).

To fix and revalidate ABAP Test Cockpit findings, as a developer user, you can run ABAP Test Cockpit checks on developed software components explicitly via ABAP Development Tools in the DEV system. See [Launching ATC Check Run from the Project Explorer](#) and [Launching ATC Check Run Explicitly](#).

If transport blocking in case of ABAP Test Cockpit findings is configured (see [Set Up Add-On Development](#)), ABAP Test Cockpit checks are executed implicitly. See [Launching ATC Check Implicitly](#).

### ATC Exemptions

If you can't clear an ABAP Test Cockpit finding by correcting the underlying problem, you can still clear it by requesting an exemption. Exemptions are created as part of a specific software component. See [Working with ATC Exemptions](#).

### Continuous Testing using ABAP Environment Pipeline

### → Tip

For in-depth information about the ABAP environment pipeline used for continuous testing, see [ABAP Environment Pipeline \[page 1328\]](#).



To schedule a regular execution of ABAP Test Cockpit checks for a software component in the test system, you can use a CI server and pipeline to automate this process. You can reuse the previously described setup of a transport from development to test system using the ABAP environment pipeline. See [Set Up Transport from Development to Test System](#).

As a DevOps engineer, configure the ABAP environment pipeline by using a static and preconfigured system. With the pipeline, the following steps are then automated, triggered by the pipeline execution of a Jenkins administrator:

- Pulling the specified software components/Git repositories
- Running the configured ABAP Test Cockpit checks

The continuous testing scenario of the ABAP environment pipeline is described in detail in [Continuous Testing on SAP BTP ABAP Environment](#).

## 4.2.10.1.5.5 Build

As an add-on administrator, you have two options for triggering the add-on build process. The recommended approach is to use the Build Product Version app in the Landscape Portal to configure and execute an add-on build pipeline, leveraging the CI/CD service on SAP BTP. Alternatively, you can implement your own pipeline manually and run it on an existing CI/CD server. In both cases, a successful pipeline execution will result in a new product version that can be installed.

The add-on build pipeline runs through multiple steps:

- For the add-on assembly, an assembly system is created in the 03 Build/Assemble subaccount. All software components that are part of the add-on product are imported into the system.

### ⓘ Note

The import in the assembly system is enabled by communication scenario [Integration \(SAP COM 0510\)](#).

- During the add-on build, delivery packages corresponding to included software component versions are created. For the add-on product version, a target vector is created and published in test scope. See [Target Vector](#).

### ⓘ Note

The build process is enabled by communication scenario [Software Assembly Integration \(SAP COM 0582\)](#).

- To ensure that the add-on provisioning works correctly, an installation test system is provisioned in the 04 Build/Test subaccount. The new product version is installed using the target vector published in the build stage.

### ⓘ Note

The installation test system uses the service plan `saas_oem`, enabling the add-on installation.

- Once the build and test installation have been completed successfully, the target vector is published in production scope.
- After a successful build, all ABAP systems used are deprovisioned (unless otherwise specified) and the add-on is technically available for deployment to the ABAP environment.

With any new add-on, it is required that you first register the add-on product and any global accounts you wish to use before building your initial product version.

→ Tip

For in-depth information about the ABAP environment pipeline used for add-on build, check out [ABAP Environment Pipeline](#).

## Prerequisites

- Using the Build Product Version app requires an existing CI/CD subscription in the same subaccount as the Landscape Portal subscription (recommended option). If you used the [Booster for Landscape Portal \[page 1358\]](#) to set up your landscape, then this is already configured.
- Using a custom pipeline configuration and Jenkins server requires the following (not recommended):
  - To execute the pipeline, you have to set up a Jenkins CI/CD server that is provisioned using the Cx server and you need an S-user in SAP One Support Launchpad with user management authorization. See [Jenkins](#), [Cx Server](#), and SAP note [1271482](#).
  - To configure the pipeline, you need to create a configuration repository and implement the add-on build scenario. See [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).

### 4.2.10.1.5.5.1 Set Up Add-On Build

The following section explains how to build the initial version of the add-on product. The result is a delivery of type AOI (Add-on Installation) for the initial release 1.0.0 of both the software component and the add-on product.

The add-on build process is orchestrated by an automated pipeline. Before the pipeline can be configured, the following steps need to be performed.

#### Add Technical Platform User to Space in Subaccount 03 Build/Assemble

As an operator, assign a technical Cloud Foundry platform user as a space developer in subaccount 03 Build/Assemble. This technical user will be used to provision the assembly system. See [Creating New Space Members and Assigning Space Developer Roles to Them](#).

ⓘ Note

If you used the [Booster for Landscape Portal \[page 1358\]](#) to set up your landscape, then this is already configured.

#### Add Technical Platform User to Space in Subaccount 04 Build/Test

As an operator, assign a technical Cloud Foundry platform user as a space developer in the space in subaccount 04 Build/Test. This technical user will be used to provision the installation test system. See [Creating New Space Members and Assigning Space Developer Roles to Them](#).

ⓘ Note

If you used the [Booster for Landscape Portal \[page 1358\]](#) to set up your landscape, then this is already configured.

## Create a Technical Communication User for Access to AAKaaS

As part of the add-on build process, the Add-on Assembly Kit is used. With AAKaaS, the on-premise tooling is now available as a cloud service that is used by the build pipeline.

To access the service that is offered via SAP Support Launchpad, a technical communication user is used. See SAP note [2532813](#).

In the *Support User Management* app in SAP ONE Support Launchpad, as an S-user, request a new technical communication user. After the technical communication user has been generated, activate the user.

### Note

To create technical communication users in SAP Support Launchpad, you have to assign the user administrator function to the logged on user.

Make sure that this technical communication user is assigned to the customer number under which the ABAP environment tenants are licensed and for which the development namespace was reserved.

Ensure that this technical communication user is assigned to the customer number under which the ABAP environment tenants are licensed and for which the development namespace was reserved.

See SAP note [2174416](#) for more information on how to create and activate technical communication users.

## Add credentials to credential store

The credentials of the three technical communication users need to be maintained in the Build Product Version app. See [Maintain Credentials](#).

If you are using your own Jenkins server, the Jenkins administrator needs to add the credentials of the three technical communication users to the Jenkins Credentials store.

## Register Add-On Product for Installation in Global Accounts

The registration of a new add-on product is a manual step. Your add-on product should only be installed in ABAP systems in your global accounts for development and production. Therefore, the product needs to be created and global accounts need to be registered with SAP using Landscape Portal.

See [Register Product](#).

## Configure ABAP Environment Pipeline

As a DevOps engineer, you have the choice between using the Build Product Version app in the Landscape Portal or creating a pipeline configuration for the add-on build scenario from scratch.

We strongly recommend using the Build Product Version app, as it greatly reduces both the initial configuration and the maintenance effort:

- the user-friendly tooling hides the complexity of configuring a pipeline definition and removes the need for an external repository,
- the app automatically delegates the execution of the pipeline to the CI/CD service on SAP BTP, removing the need for a partner-owned Jenkins server.

Creating a custom pipeline configuration allows you to modify parameters of the ABAP Environment pipeline that are not exposed in the Build Product Version app; however, the app should cover all common use cases.

Currently there are two features available in the pipeline that the app does not support:

- It is not possible to define a manual confirmation step before the target vector is published productively. This option allows you to perform integration tests in the ATI system before your final release decision. The pipeline executed by the app will simply check if the new version was installed successfully and will then automatically proceed to the publishing step.
- It is not possible to work with a permanent build system. This option allows to save on execution time, as no new system has to be provisioned. The app will always provision a new build system.

### **Build Product Version app (recommended)**

The Build Product Version app allows you to configure the pipeline directly in a Fiori UI. See [Build Product Version](#).

As a Landscape Portal Administrator, configure the Release Delivery pipeline template to be able to build your initial version. You can also configure the Test Release Delivery template in case you wish to perform a test execution, without releasing the new version.

### **Manual Configuration**

As a Jenkins administrator, create a new pipeline in Jenkins pointing to the Jenkins file of the pipeline configuration for the add-on build scenario. See [Configuration](#).

The add-on build scenario of the ABAP environment pipeline is described in detail in [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).

## **4.2.10.1.5.5.2 Build the First Add-On Version**

### **Create Maintenance Branch**

#### **→ Recommendation**

For each release and support package level, as an add-on admin, create a maintenance branch that is used for development of bug fixes and maintenance deliveries. See [Versioning and Branches \[page 1333\]](#).

In test system TST, create branch **v1.0.0** that is based on the main branch.

### **Build New Product Version**

Each version contains an add-on version number – 1.0.0 for the initial delivery – as well as the included software components, such as the corresponding names, branches, and software component versions to be used for the add-on build. You may also choose specific software component commit IDs (as seen in the Manage Software Components app) and select which languages should be included in the new product version. Add-on version numbers may not be used more than once.

### **Build Product Version app (recommended)**

New product versions for your registered add-on products can be defined directly in the Build Product Version app. The initial product version will always be of type “Release Delivery”. See [Create New Product Version](#).

The corresponding pipeline is executed automatically when the **Build Product Version** button is clicked on in the app. If the execution is successful, the initial version of the add-on will then be available for installation.

## Manual Configuration

If using a custom pipeline configuration and your own Jenkins server, create an add-on configuration file (`addon.yaml`) in the corresponding Git repository. This file includes the metadata of the add-on product that is being built, such as versioning information and the included software component versions. Please follow the best practices on how to define the `addon.yaml` file. See [Add-on Descriptor File](#).

Once the repository is updated, trigger the pipeline execution on your Jenkins server. If your pipeline includes manual confirmation steps, you will have to follow its progress and react accordingly. You may use the add-on installation test system ATI to confirm the success of the add-on installation before confirming its release. You can also use the system for additional tests, similar to the steps described in [Test in the ABAP Environment SAP Fiori Launchpad](#). For testing in a consumer-like environment, you can create tenants of type Partner Test using the Landscape Portal application. See [Use Test Tenants](#).

### → Tip

For in-depth information about versioning and branches, check out [Versioning and Branches \[page 1333\]](#).

To learn how software lifecycle management in the ABAP environment works with software components, see [Basic Concepts and Terms \[page 715\]](#).

### ⓘ Note

Please make sure that the add-on product version to be published is properly tested before confirming the release decision. This includes testing in SAP Fiori launchpad and the ABAP Test Cockpit. See [Test in the ABAP Environment SAP Fiori Launchpad \[page 1365\]](#) and [Test in the ABAP Test Cockpit \[page 1366\]](#).

### Check Add-on Build Result

Use the [Check Product Version](#) app in Landscape Portal to check whether the product version, its components, and respective packages are ready for delivery. See [Check Product Version](#).

## 4.2.10.1.6 Order and Provide

After the initial add-on version has been built, you must deploy and provide it as a SaaS solution so that the add-on can be consumed by customers via a subscription in the SAP BTP cockpit.

### 4.2.10.1.6.1 Deploy

In order to offer a multitenant application based on an ABAP add-on product, a multi-target application (MTA) is developed and deployed to the Cloud Foundry environment on SAP BTP. The MTA integrates the different services required for this scenario, such as the ABAP Solution service, responsible for provisioning ABAP systems, tenants, and users on demand. For more in-depth information, see [Multitenant Applications](#).

Once you've configured your multitenant application, you can deploy it once for testing purposes in the global account for development. After testing the subscription process during this development phase, the multitenant application can be deployed to the global account for production. See [System Landscape/Account Model](#).

After the multitenant application has been deployed for production purposes, the SaaS solution is ready for commercialization.

You have two options for implementing and deploying the application. The recommended approach is to use the Maintain Solution app in the Landscape Portal. See [Maintain Solution](#).

Alternatively, you can create your own multitenant application. This allows you to modify parameters that are not exposed in the Maintain Solution app; however, the app should cover all common use cases. See [Developing Multitenant Applications in the Abap Environment](#).

### **gCTS Delivery**

If you use gCTS instead of add-ons for delivering software components to production systems, you have to configure the multitenant application manually. In the configuration of the ABAP solution service the value for the add-on product name must be left empty. This is currently not supported by the Maintain Solution app. See [Delivery via Add-On or gCTS](#).

Prerequisites

- To configure the sizing of a SaaS solution, you have to determine the expected load per region by using the Technical Monitoring Cockpit. See [Technical Monitoring Cockpit \(Cloud Version\)](#).
- To implement and deploy a multitenant application for a SaaS solution, you have to assign the necessary entitlements in the provider subaccount, for example for the ABAP Solution service. See [Multitenant Applications](#) and [Entitlements and Quotas](#).

## **4.2.10.1.6.1.1 Sizing**

A multitenant application has some central sizing properties that dictate the scope and metric of your offering, or in other words, how many resources will be available for each consumer.

*Tenant Limit:* Before deploying your application, you must decide on a tenant limit. Once that number of consumer tenants is reached in a system, a new system will be provisioned for the next consumer. If you choose a limit of 1, you will be defining a single tenant offering with one consumer per system.

### → Tip

For in-depth information about multitenancy, see [this section](#).

Additionally, you need to define the sizing of each provisioned ABAP system. The sizing is defined by the number of ABAP compute units (runtime) and HANA compute units (persistence) each system is assigned. These values are reflected by the parameters `size_of_runtime` and `size_of_persistence`. For more information, see [Creating an ABAP System \[page 180\]](#).

For multitenancy offerings, there's no sizing/quota per customer. You must decide on an overall sizing depending on the expected load in a region. You can use the Technical Monitoring Cockpit to assist you in this.

### ⓘ Note

If the quota for a system is exceeded, you can request a resizing of the ABAP runtime or persistency depending on the needs of the SaaS application.

## 4.2.10.1.6.1.2 Using the Maintain Solution App

### Configure the solution

As a Landscape Portal administrator, you can implement the multitenant application using the *Maintain Solution* app in the Landscape Portal. Create a new solution using:

- the previously registered product,
- the product version that shall be installed initially (“latest” will always install the newest available product version),
- the usage mode “prod”, unless you do not intend to deploy a productive application at the end,
- the consumer tenant limit and system sizing values that you have decided on.

For more information regarding solutions, see [Create Solution](#).

### Deploy the solution for test purposes

After creating the solution, it is recommended to perform a test deployment to validate the subscription process. Use the *Maintain Solution* app to create a deployment configuration for your solution:

- Target the Provide space in the O5 Provide subaccount in your global account for development. You need to specify the corresponding CF API Endpoint, CF Organization Name and CF Space Name.
- Choose the domain type “shared”. Since this is not a productive deployment, it is sufficient to use one of the available app domains provided by SAP.
- Since you are deploying to a shared app domain, a unique route prefix is needed to uniquely identify your application endpoint.
- Maintain the subdomain of the subaccount where you intend to test the subscription process.

For more information regarding deployment configurations, see [Create Deployment Configuration](#).

### Test the solution

You can test the multitenant application by subscribing from a consumer subaccount created in the global account for development. See [Subscribe to Multitenant Applications Using the Cockpit](#).

Once the subscription is successful, you may also want to test the initial user onboarding process. First, assign the onboarding role collection, which includes the SolutionAdmin role, to your user. Afterwards, access the application via the consumer subaccount and trigger the user onboarding. For more details, please see the section on the productive tenant onboarding.

## Deploy the solution for production purposes

After successfully testing the subscription process you can proceed to a productive deployment of your solution. Use the [Maintain Solution](#) app to create a second deployment configuration for your solution:

- Target the Provide space in the 05 Provide subaccount in the global account for production. You need to specify the corresponding CF API Endpoint, CF Organization Name and CF Space Name.
- Choose the domain type “custom”. It is recommended to use a custom domain for your productive application, such that consumers can access it via a recognizable URL.
- It is recommended to use a wildcard route in conjunction with custom domains. This will route all requests towards that domain to your application.

To set up a custom domain, please follow the steps outlined in [Configuring Custom Domains](#) or [Get Started with the Custom Domain Manager](#). After registering the custom domain in your provider subaccount, it becomes available as a private domain (see [Private domains](#)) for route creation in any of the Cloud Foundry spaces of the subaccount. Once you have created the production deployment configuration, you can trigger it using the [Deploy](#) action, see [Deploy Solution](#). Once deployed, your solution should appear in the service marketplace on SAP BTP Cockpit for subaccounts within your global account for production.

### 4.2.10.1.6.1.3 Configuring a Solution Manually

You can configure the multitenant application manually, which allows you to finetune your solution beyond the options offered in the [Maintain Solution](#) app. In this case, you must configure the individual project files and deploy the project using the corresponding command line tools.

In the following sections, the general steps are described. For a description of multitenant applications in general and more detailed configuration options, see [Multitenant Applications \[page 1376\]](#).

#### Configure the solution

It is recommended that you initially configure your solution using the Maintain Solution app and then use the Download functionality to obtain the application descriptor file. Since the download is available on deployment configuration level, it is necessary to create both a solution and a deployment configuration.

Follow the steps detailed in Using the Maintain Solution app (see [Order and Provide \[page 1371\]](#)), creating both the solution and the deployment configuration for the test phase. Once configured, use the Download function to obtain the application descriptor. Finally, clone the reference solution linked below and replace the standard descriptor with the generated one. This results in a project that is already configured, making it the perfect starting point to implement your modifications.

The reference solution provided by SAP can be found here: [GitHub - sap-software/abap-saas-reference-solution: Template of an ABAP SaaS Reference Solution allowing the SAP Partner to deploy a solution containing approuter and XSUAA dependencies](#).

### Note

gCTS Delivery: If you use gCTS instead of add-ons for the delivery of software components to production systems, the value for the add-on product name in the application descriptor has to be empty. See [Delivery via Add-On or gCTS \[page 1342\]](#).

## Deploy the solution for test purposes

After configuring the multitenant application, it is recommended to perform a test deployment to validate the subscription process.

Your project should already have the correct deployment configuration, as you maintained this in the Maintain Solution app before generating the application descriptor. As a result, you can directly build and deploy your project.

Use the [Cloud MTA Build Tool](#) and the [Cloud Foundry CLI](#).

Once deployed, your solution should appear in the service marketplace on SAP BTP Cockpit for subaccounts within your global account for development.

## Test the solution

You can test the multitenant application by subscribing from a consumer subaccount created in the global account for development. See [Subscribe to Multitenant Applications Using the Cockpit](#).

Once the subscription is successful, you may also want to test the initial user onboarding process. First, assign the onboarding role collection, which includes the SolutionAdmin role, to your user. Afterwards, access the application via the consumer subaccount and trigger the user onboarding. For more details, please see the section on the [Initial User Onboarding \[page 1395\]](#).

## Deploy the solution for production purposes

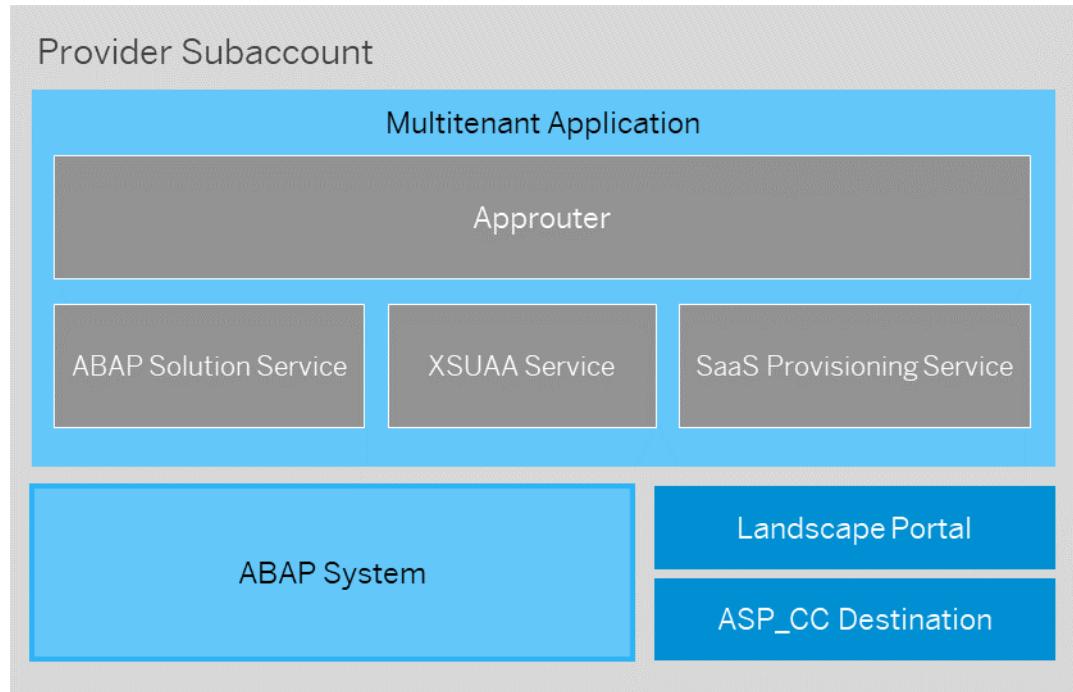
After successfully testing the subscription process, you can proceed to a production deployment of your solution.

You can use MTA extension descriptors to configure the required values for deployment to different targets. This allows you to deploy the same project to different subaccounts. See [Multitenant Applications \[page 1376\]](#).

Build and then deploy your project. Once deployed, your solution should appear in the service marketplace on SAP BTP Cockpit for subaccounts within your global account for production.

#### 4.2.10.1.6.1.4 Multitenant Applications

To make the software components in the ABAP System available for multiple different consumers, a multitenant application is deployed into the provider subaccount. This application can then be accessed by different tenants through a dedicated URL.



For the multitenant application, different services and apps are used:

- **Approuter application:** The application router is the single point-of-entry for the multitenant application and used to forward tenant-specific requests to the corresponding tenant in the ABAP system. See [Application Router](#).
- **ABAP Solution Service:** ABAP Solution service is required to create multitenant-enabled, ABAP environment-based SaaS solutions. Instead of a service binding from the approuter application to the ABAP environment business service, the approuter is bound to an ABAP Solution service instance. The service orchestrates the creation of ABAP systems upon consumer subscription as well as tenant provisioning/decommissioning. Requests routed via the approuter application to the ABAP solution service instance are automatically proxied to the corresponding ABAP tenant.
- **SAP Authorization and Trust Management service (XSUAA):** The xsuua service instance acts as an OAuth 2.0 client to the multitenant application, and to the ABAP Solution service instance. The SAP Authorization and Trust Management service lets you manage user authorizations and trust to identity providers used for the multitenant application. The service instance is created with a corresponding security configuration. See [SAP Authorization and Trust Management Service in the Cloud Foundry Environment](#).
- **SaaS Provisioning Service:** The SaaS Provisioning service allows application providers to register multitenant applications and services in the Cloud Foundry environment in SAP Business Technology Platform. After a SaaS Provisioning service instance is created, the multitenant application becomes available for subscription in a consumer subaccount.
- **ABAP System:** ABAP systems and tenants used to provide the multitenant application are not created manually but are managed by the ABAP Solution service: Upon first subscription the ABAP system is

created as well the consumer tenant in the system. To be able to create the system, the ABAP solution service uses credentials of a technical platform user that is defined in the ASP\_CC destination. Depending on the configuration of the ABAP solution, for every subsequent subscription either a new tenant is created in the same ABAP system (`consumer_tenant_limit > 1`) or a new system is created for every new subscription (`consumer_tenant_limit = 1`). A new ABAP system is also created if all the existing ones are "full" because the consumer tenant limit that you specified was reached. Note that an ABAP system can only be deleted if no consumer tenants exist.

Consumer subscription-based multitenancy is only possible in those systems in which you provide your software as a service (SaaS) solution to consumers, i.e. your production and test systems. It is not available in development systems.

See [Creating an ABAP System](#).

- **Landscape Portal:** The Landscape Portal acts as a central tool to allow service providers to perform lifecycle management operations such as add-on updates, provisioning new consumers as new tenants, and more. A subscription to the Landscape Portal in the globalaccount where the multitenant application is deployed is required for the subscription flow to work.  
See [Landscape Portal](#).
- **ASP\_CC Destination:** An OAuth2Password destination pointing to the Cloud Foundry Cloud Controller API is required for the ABAP Solution service to create and access ABAP service instances. Credentials of a technical platform user, that is assigned as a space developer in the space where the multitenant application is deployed, are maintained.

## Multitarget Application

The multitenant application is deployed as a multitarget application (MTA), which is logically a single application comprised of multiple parts created with different technologies.

The MTA describes the desired result in an `mta.yaml` file which may be extended with `mta` extensions (`.mtaext`) using the MTA model which contains MTA modules, MTA resources and interdependencies between them. Afterwards, the MTA deployment service validates, orchestrates, and automates the deployment of the MTA, which results in Cloud Foundry (CF) applications, services and SAP-specific contents. For more information about the Multitarget Application model, see the official [The Multitarget Application Model specification](#). For more details on how to use it in the CF Environment, which is the case here, see [Multitarget Applications in the Cloud Foundry Environment](#).

A complete reference example of the multitenant application MTA is provided in GitHub/SAPSamples: [ABAP SaaS Reference Solution for ABAP solution provider](#) ↗

## MTA Parameters

Parameters are defined at the beginning of the `mta.yaml` file, so that the actual structure of the MTA model does not have to be changed but can be extended via extension files. The parameters are referenced in the subsequent sections of the MTA using the syntax  `${parameter-name}`. For more details on parameters, see [Parameters and Properties](#).

- **Freely defined parameters:**
  - `app-domain`: The domain that should be used for the approuter and therefore the SaaS application. In our example, this is the  `${default-domain}` (equals to the `cfapps-domain` of the current landscapes) for a development deployment; for production it should be the custom domain (so here: `mydomain.com`).
  - `appname`: The value that should be used as `xsappname` and solution name (here: `abap-saas-reference-solution`).
  - `route-prefix`: The value that should be used as a prefix in the route of the approuter, allowing it to deploy the same MTA to the same landscape multiple times using the default landscape domain.

- addon-product-name: Name of add-on product to be installed
- provider-admin-email: Provider administrator e-mail address used for provisioning of ABAP systems
- saas-display-name: Display name of SaaS application shown in subscription tile
- saas-description: Description of SaaS application shown in subscription tile
- tenant-mode: Whether to provide a 'single' or 'multi' tenancy solution
- **Parameter with semantics to MTA deployer:**
  - enable-parallel-deployments: If set to `true`, deploys multiple modules at the same time.

## MTA Extension Descriptors

The parameters defined in the MTA can be adapted for the corresponding landscape using MTA extension descriptor files. These files extend the original MTA descriptor file (`mta.yaml`) and provide certain deployment specific configurations/parameters/values. The descriptor files have the file extension `*.mtaext` and are used during the deployment using CF CLI with MultiApps plugin, see [MultiApps CF CLI Plugin](#).

For more information on MTA extension descriptors, see: [Defining MTA Extension Descriptors](#).

The extension descriptor files are usually in the folder “extensions” next to the `mta.yaml` file.

Depending on the stage of implementation of the multitenant application, a different approuter configuration is used. This is controlled via different extension descriptor files used during MTA deployment:

- Configuration for **Development Phase**:

For development, test and demo purposes, MTA deployment can be done using the  `${default-domain}` as `app-domain` and a `route-prefix`, which will avoid collisions due to the default domain. Additionally, we suggest setting the property `XS_APP_LOG_LEVEL` on the approuter module to `debug` to ensure that there is more detailed logging output.

- Configuration for **Production Phase**:

For purpose of providing a solution / solutions to applications consumers in a production scenario, the MTA deployment should be done using a custom domain configured with the `app-domain` parameter (`app-domain`) and an empty route-prefix (`route-prefix.`). Using the custom domain, a wildcard route for tenant access ("`* .${app-domain}`") can be created.

## MTA Build and Deploy

Before MTA deployment, the MTA project is built using the [Cloud MTA Build Tool \(MBT\)](#). Afterwards the deployment-ready multitarget application archive (`*.mtar` file) is deployed in combination with the corresponding MTA extension descriptor file using the Cloud Foundry CLI with the MultiApps plugin: `cf deploy <path to .mtar file> -e <path to MTA extension descriptor>`

## Related Information

[Installing the cf CLI](#)

[MultiApps CF CLI Plugin - Download and Installation](#)

[MultiApps CF CLI Plugin - Usage](#)

## 4.2.10.1.6.1.4.1 ABAP Solution Service

### ↔ Sample Code

```
- name: abap-solution
  type: org.cloudfoundry.managed-service
  parameters:
    service: abap-solution
    service-plan: standard
    service-name: ${appname}-abap-solution
  config:
    name: ${appname}
    consumer_id_pattern: ${consumer-id-pattern}
  plans:
    - plan_name: ${plan-name}
      addon_product_name: ${addon-product-name}
      size_of_runtime: ${size-of-runtime}
      size_of_persistence: ${size-of-persistence}
      provider_admin_email: ${provider-admin-email}
      consumer_tenant_limit: ${consumer-tenant-limit}
      sap_system_name: ${sap-system-name}
      usage: ${usage-mode}
  xs-security:
    xsappname: xs-app-${appname}
```

ABAP Solution Service is used to enable multitenancy for ABAP Environment systems: Instead of creating the ABAP system manually it is created by the ABAP Solution service during the first subscription of an application consumer or once the consumer tenant limit for all viable systems has been reached.

Credentials provided in the ASP\_CC destination are used by ABAP Solution Service to authenticate at the Cloud Foundry Cloud Controller API endpoint and to create required ABAP service instances.

The service plan used for created ABAP service instances is depending on the `addon_product_name` parameter: `abap/standard` is used for a system without add-on, `abap/saas_oem` is used for a system with installed add-on. Provisioning parameters like `sap_system_name`, `provider_admin_email` and sizing-relevant parameters can be defined.

After a new system is created, a tenant for the application consumer is provisioned. The business type (see [Tenant Business Types](#)) used for provisioning of the tenant depends on the `usage` parameter: In case of `usage = test`, the tenant is created as `Partner Customer Test` tenant. Whereas with `usage = prod`, the tenant is created as `Partner Customer Production` tenant. Parameter `tenant_mode` controls if per subscription a new tenant is created in the same system and with `consumer_tenant_limit` the number of tenants per system is configured.

### ABAP System: saas\_oem plan

This service plan is like the standard service plan of the ABAP service (see [Create an ABAP System](#)), but it is used as "backing service" for a partner-provided SaaS solution. In comparison to a standard service plan of the ABAP service, there is one main differences: An add-on product (see [The Add-on product](#) ) which is installed during system provisioning can be specified.

- Parameter: `addon_product_name`  
(specifies which ABAP add-on to install during provisioning)
- Parameter: `addon_product_version`  
(if - for testing purpose - another add-on version instead of the most recent released version shall be installed)

## ABAP Solution Service Parameters

Name	Data Type	Description	Purpose	Updateable
name	string	Name of the solution as defined by the provider. Must be unique in the scope of the Global Account (or globally if this information is not available)	Used to identify the solution and will be passed through to ABAP System (saas_oem plan)	No
sap_system_name (optional)	string	Name of the system as defined by the provider. If a value is supplied, the new system(s) will be created with this parameter.	Passed through ABAP System	No
addon_product_name (optional)	string	Registered name of the add-on product	Passed through ABAP System (saas_oem plan)	No
addon_product_version (optional)	string	Version of the add-on product to be installed. A released product version needs to be defined. If you do not specify the parameter, the latest released product version will be used during the add-on installation. If you specify other versions than allowed, the add-on installation and therefore also the system provisioning and consumer subscription will be unsuccessful.	Passed through ABAP System (saas_oem plan)	Yes (updated value applies for new systems only)

Name	Data Type	Description	Purpose	Updateable
consumer_tenant_limit	int	<p>Maximum number of consumer tenants in the multitenant ABAP system. If the consumer tenant limit is exceeded, a new multitenant system will be created.</p> <p>consumer_tenant_limit &gt;= 1</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>A recently deleted tenant will be counted towards the limit, in case it is restored during the grace period. After the 30-day grace period is up, it will no longer be considered for the limit.</p> </div>	Passed through ABAP System (saas_oem plan)	Yes
size_of_runtime	int	Default sizing (ABAP runtime) of abap system(s) created/provisioned by solution	Passed through to ABAP System	Yes (updated value applies for new systems only)
size_of_persistence	int	Default sizing (ABAP persistence [HANA]) of abap system(s) created/provisioned by solution	Passed through to ABAP System	Yes (updated value applies for new systems only)

Name	Data Type	Description	Purpose	Updateable
tenant_mode (deprecated)	enum (string): - single - multi	Tenant Mode of the solution	Decides whether a customer will have a tenant in - a dedicated system (single) - a shared system (multi)  <b>single:</b> will trigger a system creation and a tenant onboarding  <b>multi</b> will trigger a system creation (only if necessary) and a tenant onboarding  The parameter will be passed through to ABAP System (saas_oem plan).	No
consumer_id_pattern	string (regex)	String containing a regular expression with a capturing group. The <b>subdomain of the consumer</b> is matched against this regular expression. The value of the first capturing group is used as consumer id.	To allow the provider to group his consumer subaccounts based on a self-chosen consumer identifier: e.g. the subdomains of consumer subaccounts always include a static part that identifies the application consumer, then event if there are multiple test & production consumer subaccounts, the application consumer can be identified in Landscape Portal.	No
provider_admin_email	email	Email address of initial provider user	Passed through to ABAP System (saas_oem plan).	Yes

Name	Data Type	Description	Purpose	Updateable
xs-security/xsapp-name (optional)	tring	<p>xsappname used for the OAuth clone during instance creation.</p> <p>Will be visible in the security section of SAP BTP Cockpit when assigning the initial on-boarder role. Will be the "Application Name" shown in the Roles UI.</p> <p><b>Default:</b> Service Instance GUID</p> <p><b>Recommendation:</b> Provide the same value as defined in the xs-security.json of the XSUAA Instance of your application</p>	Allows providing a meaningful application name to assign the on-boarding roles.	No
usage	enum (string): - test - prod	Specifies whether it is a test or production solution.	In case of usage = test a Partner Customer Test tenant is created, whereas with usage = prod, a Partner Customer Production tenant is provisioned.	No

## 4.2.10.1.6.1.4.2 SAP Authorization and Trust Management Service

### Sample Code

```

- name: xsuaa
  type: com.sap.xs.uaa
  requires:
    - name: abap-solution
  parameters:
    service-plan: application
    service-name: xsuaa
  config:
    oauth2-configuration:
      redirect-uris:
        - "https://*.${app-domain}/**"
    xsappname: ${appname}
    tenant-mode: shared
    scopes:
      - name: $XSAPPNAME.Callback
        description: With this scope set, the callbacks for tenant onboarding, offboarding and getDependencies can be called.
    grant-as-authority-to-apps:

```

```

        - $XSAPPNAME(application,sap-provisioning,tenant-onboarding)
foreign-scope-references:
  - uaa.user
role-collections:
  - name: ${appname}-admin
    role-template-references:
      - $XSSERVICENAME(${appname}-abap-solution).SolutionAdmin

```

The xsuaa service instance, acts as an OAuth 2.0 client to the multitenant application, and to the ABAP Solution service instance. It provides access to the SaaS Provisioning service for calling callbacks and getting the dependencies API by granting corresponding scopes.

A role collection is defined as part of the xsuaa configuration which provides authorization for the initial user onboarding in the new ABAP tenant after a subscription was done.

For this purpose, the “SolutionAdmin” role is referenced. See [Quick Start: Create Role Collections \(with Predefined Roles\)](#).

Finally, a foreign scope reference is defined to ensure a token exchange will be possible and the ABAP systems can be called.

### 4.2.10.1.6.1.4.3 Saas Provisioning Service

#### ↔ Sample Code

```

- name: saas-registry
  type: org.cloudfoundry.managed-service
  parameters:
    service: saas-registry
    service-plan: application
    service-name: saas-registry
  config:
    xsappname: ${appname}
    appName: ${appname}
    appUrls:
      getDependencies: https://cis${route-prefix}.${app-domain}/callback/v1.0/dependencies
      onSubscription: https://cis${route-prefix}.${app-domain}/callback/v1.0/tenants/{tenantId}
    displayName: ${saas-display-name}
    description: ${saas-description}

```

The SaaS Provisioning service allows application providers to register multitenant applications and services in the Cloud Foundry environment in SAP Business Technology Platform.

#### SaaS Provisioning Service Parameters

xsappname

The xsappname configured in the security descriptor file used to create the XSUAA instance.

getDependencies	<b>(Optional)</b> Any URL that the application exposes for GET dependencies. If the application does not have dependencies and the callback is not implemented, it should not be declared. <b>Note:</b> The JSON response of the callback must be encoded as either UTF8, UTF16, or UTF32, otherwise an error is returned. Important: You can either provide your own getDependencies Callback or use the default implementation of the AppRouter (recommended if no special logic is needed). <b>But:</b> If an own implementation is provided you have to make sure that the ABAP Solution instance is returned as a dependency.  The path is: /callback/v1.0/dependencies
onSubscription	Any URL that the application exposes via PUT and DELETE subscription. It must end with /{tenantId}. The tenant for the subscription is passed to this callback as a path parameter. You must keep {tenantId} as a parameter in the URL so that it is replaced at real time with the tenant calling the subscription. This callback URL is called when a subscription between a multitenant application and a consumer tenant is created (PUT) and when the subscription is removed (DELETE). <b>Important:</b> You can either provide your own onSubscription Callback or use the default implementation of the approuter (recommended if no special logic is needed).  The path is: /callback/v1.0/tenants/{tenantId}
displayName	<b>(Optional)</b> The display name of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's technical name.
description	<b>(Optional)</b> The description of the application when viewed in the cockpit. For example, in the application's tile. If left empty, takes the application's display name.
category	<b>(Optional)</b> The category to which the application is grouped in the Subscriptions page in the cockpit. If left empty, gets assigned to the default category.

## Callback APIs

Callback URLs defined in parameters `getDependencies` and `onSubscription` are called during the subscription flow and need to be callable via a consumer-independent URL. For this purpose, a dedicated URL that matches the `TENANT_HOST_PATTERN` of the approuter application should be defined as part for the MTA project.

Depending on the phase of the multitenant application, a different `TENANT_HOST_PATTERN` is defined:

- **During development phase:**

`TENANT_HOST_PATTERN: ^(.*)-abap-saas-app.cfapps.eu10.hana.ondemand.com`

Define a route `cis-abap-saas-app.cfapps.eu10.hana.ondemand.com` for SaaS Provisioning service callback URLs

- **During production phase:**

`TENANT_HOST_PATTERN: ^(.*) .mydomain.com`

Define a route cis.mydomain.com for SaaS Provisioning service callback URLs

## Subscription Management Dashboard

With Subscription Management Dashboard as the dashboard UI of SaaS Provisioning Service, the application provider can centrally manage subscriptions, e.g. unsubscribe existing application consumer subscriptions. See [Using the Subscription Management Dashboard](#).

## 4.2.10.1.6.1.4.4 Approuter Application

To authenticate business users of the application at runtime, a tenant-aware approuter application and the xsuaa service in SAP Business Technology Platform are used. This will also provide the necessary callbacks for the SaaS Provisioning Service.

The enhanced approuter application is as a logical part of the multitenant application and configured as an external access point of the application.

For more details on the application router, see [Application Router](#).

With the middleware @sap/asp-middleware component a library (npm module) is provided that enhances the approuter to also take care of automatically routing the consumer to their ABAP tenants. See [@sap/asp-middleware](#).

### Approuter Configuration

To make sure that the approuter routes all relevant requests to the ABAP Solution, a routing configuration is configured:

#### Sample Code

```
{  
  "authenticationMethod": "route",  
  "welcomeFile": "/ui",  
  "logout": {  
    "logoutEndpoint": "/sap/public/bc/icf/logoff",  
    "logoutPage": "/ui"  
  },  
  "routes": [  
    {  
      "source": "^/sap/(.*)$",  
      "target": "/sap/$1",  
      "authenticationType": "xsuaa",  
      "service": "com.sap.cloud.abap.solution",  
      "csrfProtection": false  
    },  
    {  
      "source": "^/ui(.*)$",  
      "target": "/ui$1",  
      "authenticationType": "xsuaa",  
      "service": "com.sap.cloud.abap.solution",  
      "csrfProtection": false  
    }  
  ]  
}
```

- **Requests to the .../sap/... path** are backend requests and will be routed to the ABAP Solution service after successful authentication. CSRF token protection is disabled for this route as CSRF protection is already enforced via the underlying services in the ABAP system.
- **Requests to the .../ui/...** are requests to the Fiori Launchpad / End User UI and will be routed to the ABAP Solution service after successful authentication. CSRF token protection is disabled for this route as CSRF protection is already enforced via the underlying services in the ABAP system.

For more details on the parameters, see [Routing Configuration File](#).

## Approuter Multitenancy

The consumer accesses their multitenant application via the tenant-specific url which is resolved by the app router and "forwarded" (calls) to the corresponding abap tenant (client). The URL follows the same pattern for all application consumers and differs depending on the implementation phase of the multitenant application:

- **During development phase:**

```
<SUBSCRIBER_TENANT_SUBDOMAIN>-<APPROUTER_APPLICATION_HOST>. <SAP-
 PROVIDED_STANDARD_DOMAIN>
```

In each development landscape, SAP provides a different standard domain you can use to create URLs. Using that domain, and following the specified URL structure, an example for a created URL could be: tenant1-myapprouter.cfapps.eu10.hana.ondemand.com

Here `tenant1-myapprouter` is the hostname, and `cfapps.sap.hana.ondemand.com` is the SAP-provided standard domain for this development landscape. In this format, a new route must be created manually for each new tenant.

The common pattern identifying tenant-specific URLs is defined with `TENANT_HOST_PATTERN: ^(.*)->-<APPROUTER_APPLICATION_HOST>. <SAP- PROVIDED_STANDARD_DOMAIN>`

- **During production phase:**

```
* .<YOUR_CUSTOM_DOMAIN>
```

A custom domain is registered via the custom domain service, then a pattern with \* as the wildcard hostname can be established.

In this format, since the wildcard is replaced by an actual tenant during runtime, there is no need to create a new route for each new tenant manually.

The common pattern identifying tenant-specific URLs is defined with `TENANT_HOST_PATTERN: ^(.*)-<YOUR_CUSTOM_DOMAIN>`

For more information, see [Using Custom Domains](#).

The application router derives the tenant from the URL and calls the tenant-aware XSUAA, which is responsible for the authentication of the business user. The UAA reads the tenant and gets the customer-specific identity provider (IdP) from the tenant-specific identity zone. Then, the XSUAA delegates the authentication to the configured IdP, and creates a JSON Web Token (JWT) that contains the tenant, the current user, and the authorization scopes of the user. The JWT is then sent back to the application router, and from there to the application.

To read and store tenant-specific data, the multitenant application needs to know the tenant ID. Using the Container Security API, the multitenant application is enabled to read the tenant, user, and scopes from the given JWT. The API also validates whether the JWT is valid. The tenant information is contained in the `identityZone` user attribute.

For more information, see:

[Web Access in the Cloud Foundry Environment](#)

[What Is Authorization and Trust Management](#)

## 4.2.10.1.6.1.4.5 Cloud Controller Access Destination

A destination called ASP\_CC needs to be created for ABAP Solution Service to be able to access the Cloud Foundry Cloud Controller API for creation and access to ABAP service instances.

### Prerequisites

- You've created a user in SAP ID service.
- The user has been added as a Cloud Foundry org member in the provider subaccount where the Cloud Foundry environment is enabled.
- The User has been added with the role of a Space Developer, in the space where the multitenant application (MTA) is deployed.

#### ⓘ Note

Please note that you should never set personal credentials (e.g. for your account) in the User and Password fields. Always use a technical user instead.

### Process Steps

1. In the provider subaccount navigate to Connectivity -> Destinations
2. Create a new destination as follows:

Property	Value
Name	ASP_CC
Type	HTTP
Description	Access the Cloud Foundry Cloud Controller API
URL	< API Endpoint domain of Cloud Foundry landscape> You can find it under your subaccount > Overview.

#### ⚡ Example

<https://api.cf.eu10.hana.ondemand.com>

Proxy Type	Internet
Authentication	OAuth2Password
User	<User ID/E-mail>
Password	<Password>
Client ID	cf
Client Secret	<needs to be left blank>

Property	Value
Token Service URL	http:// uaa.<landscape-host>/oauth/token
<b>Example</b> https://uaa.cf.eu10.hana.ondemand.com/oauth/token	

For more information on landscapes/regions, see [Regions and API Endpoints for the ABAP Environment](#).

Instead of creating the ASP\_CC destination manually, you can also save the following as a text file instead and then import the destination as a template:

#### ↔ Sample Code

```
Type=HTTP
      clientId=cf
      Authentication=OAuth2Password
      Name=ASP_CC
      tokenServiceURL=https\://
uaa.cf.eu10.hana.ondemand.com/oauth/token
      ProxyType=Internet
      URL=https\://api.cf.<e.g.eu10>.hana.ondemand.com
```

## 4.2.10.1.6.1.4.6 Regions

As a provider, you can deploy your SaaS solution in a subaccount that is located in one of the regions available for the ABAP environment, see [Regions and API Endpoints for the ABAP Environment \[page 37\]](#).

A consumer can then subscribe to and access your SaaS solution from a subaccount. The consumer subaccount needs to be located in the same region as the subaccount which you deployed the SaaS solution in.

Keep in mind that regions are chosen on the subaccount level: For each subaccount, you select exactly one region. This is done when creating the subaccount. The **Landscape Portal** can be accessed from subaccounts located in regions that are available for the ABAP environment. You thus need to make sure that the subaccount from which you want to access the **Landscape Portal** has the correct region attribute (that is, the technical key, such as cf-eu10). For more information on regions, see [Regions and Accessing the Landscape Portal](#).

## 4.2.10.1.6.1.4.7 Troubleshooting

### Multitenant application cannot be undeployed

- Always use MTA Undeployment (cf undeploy <MTA ID> --delete-services) instead of deleting service instances individually.

- Check if the saas-registry service instance still contains active subscriptions: the saas-registry instance can only be deleted once there are no more active subscriptions. Unsubscribe using the [Subscription Management Dashboard](#). Note, however that if the subscription is forced to be deleted through any additional channel (using the Subscription Management Dashboard or by subaccount deletion), it will lead to an orphan tenant in the ABAP Solution. Providers need to be aware of this before deleting subscriptions with the 'force' option.

## Cannot create consumer-specific route according to `TENANT_HOST_PATTERN` (in the development phase)

### ↔ Sample Code

```
TENANT_HOST_PATTERN: (*.*)-my-consumer-subdomain-productabcdefgijklmnopq-saas-
solution-dev.cfapps.eu10.hana.ondemand.com
Consumer Subaccount Subdomain: my-consumer-subdomain
Route Domain: cfapps.eu10.hana.ondemand.com
Route Hostname: my-consumer-subdomain-productabcdefgijklmnopq-saas-solution-
dev (64 characters)
```

## After deployment of MTA for multitenant application, no abap system is created

- ABAP systems are automatically created by the ABAP Solution Provider (ASP) during the first subscription and not after the solution deployment.

## Issues with parameter `consumer_tenant_limit`

- Keep in mind that recently deleted consumer tenants are counted towards the tenant limit as long as they are still in retention time. Systems are automatically created. Non-consumer tenants (provider tenant, test tenants) do not count towards this limit.

## 4.2.10.1.6.1.4.8 Using a Custom Domain for the Multitenant Application

The SAP Custom Domain Service (see [SAP Custom Domain service](#)) lets you configure your own custom domain (see [What Is Custom Domain?](#)) to publicly expose your application instead of using an SAP-provided default/shared/standard domain like cfapps.eu10.hana.ondemand.com for your application URLs.

By using this service, subaccount owners can make their SAP BTP applications accessible via a custom domain that is different from the default one; for example, myshop.com.

In case of multitenant applications (see [Multitenant Applications \[page 1376\]](#)), using a custom domain comes with another advantage: For each subscription there needs to be a CF route (see [Routes](#)) that is mapped to the approuter module of the multitenant application, so that the application is accessible via a URL.

While using a default domain, a new route like <SUBSCRIBER\_TENANT\_SUBDOMAIN>-<APPROUTER\_APPLICATION\_HOST>. <SAP-PROVIDED\_STANDARD\_DOMAIN> must be created manually for each new tenant.

By configuring the multitenant application with a custom domain, a route with a wildcard hostname like \* .<YOUR\_CUSTOM\_DOMAIN> can be mapped, so that there is no need to create a new route for each new tenant manually. For more information, refer to [Approuter Application \[page 1386\]](#).

It is recommended to use a custom domain for your multitenant application only during the production phase; during the development phase using a default domain is sufficient.

Please follow the steps outlined in [Configuring Custom Domains](#) or [Get Started with the Custom Domain Manager](#) (refer to PaaS-User) to register your custom domain in the provider subaccount's CF org (see [Multitenancy in the ABAP Environment \[page 1407\]](#)). After registering the custom domain successfully, the domain becomes available as private domain (see [Private domains](#)) for route creation in any space of the provider subaccount's CF org.

To create a multitenant application for use with a custom domain, following configurations are adjusted accordingly:

- **TENANT\_HOST\_PATTERN** configuration of the approuter application: String containing a regular expression with a capturing group. The request host is matched against this regular expression. The value of the first capturing group is used as tenant id. Please refer to [TENANT\\_HOST\\_PATTERN](#).
- SaaS Registry Callback URLs `getDependencies` and `onSubscription`. Please refer to [SaaS Provisioning Service \[page 1384\]](#).
- XSUAA OAuth 2.0 Client configuration redirect-uris. Please refer to [Listing Allowed Redirect URIs](#)
- **CF routes** assigned to approuter application

For the subscriptions to the multitenant application to work correctly, configured SaaS Registry callback URLs need to be reachable (cis... route). For Application Routing routing to work, the approuter needs to be able to resolve the application URL based on current **TENANT\_HOST\_PATTERN** configuration.

You can use the [Maintain Solution](#) app in the Landscape Portal to easily create such a multitenant application with custom domain, please refer to [Create Deployment Configuration → Step 3: Routing Information](#).

Or, if you are following the MTA-based approach to create your multitenant application, in the [MTA Extension descriptor for production phase](#) these configurations are changed using following MTA parameters:

- **app-domain**: the registered custom domain, e.g. mydomain.com
- **route-prefix**: needs to be empty because application URLs are mapped using following structure:  
\* .<YOUR\_CUSTOM\_DOMAIN> (dot separator) and therefore do not include a route prefix

A route with wildcard hostname for tenant access is added to the approuter module, on top of the route used for SaaS Registry callback URLs:

### ↔ Sample Code

```
modules:
  - name: approuter
    parameters:
      routes:
        - route: "cis${route-prefix}.${app-domain}"
```

```
- route: "*${route-prefix}.${app-domain}"
```

To reduce risks use [Blue-Green Deployment of Multitarget Applications](#) while deploying multitenant application MTA during production phase.

After MTA build and deployment, the approuter application is created with a route cis.mydomain.com that is used for the SaaS Registry callback URLs `getDependencies` and `onSubscription`. On top of that a route with wild card hostname `*.mydomain.com` is mapped to the approuter. In this format, since the wildcard is replaced by an actual tenant during runtime, there is no need to create a new route for each new tenant manually.

If you change the domain type of your multitenant application from shared to custom domain, you need to update the application URL of existing subscriptions. Please use the option to update the application URL of an active subscription in the [Subscription Management Dashboard](#). In such case, make sure to inform application consumers beforehand about temporary downtimes that might occur during this change.

## 4.2.10.1.6.2 Commercialize

To offer your SaaS solution to customers, register the SaaS solution in the SAP Store. See [SAP Store](#).

Additionally, you can request an SAP Extension Suite certification for your product so that it gets listed in the SAP Certified Solutions Directory. See [SAP Extension Suite Certification](#) and [SAP Certified Solutions Directory](#).

After finishing this process, your customers can order the SaaS solution, which is the preliminary step to actual subscription.

### Prerequisites

Before registering a SaaS solution in SAP Store or certifying the solution, you have to deploy the first add-on version and multitenant application. See [Multitenant Applications \[page 1376\]](#).

### 4.2.10.1.6.2.1 (Optional) Register SaaS Solution in SAP Store

SAP Store is the enterprise marketplace where we bring together customers and partners on a single, easy-to-use, global online platform. See [SAP Store](#).

To learn more about how you can offer your SaaS solution to SAP customers via SAP Store, see [Getting Started with SAP Store](#).

## 4.2.10.1.6.2.2 (Optional) Certification

With an SAP Extension Suite certification, your SaaS solution is featured in the SAP Certified Solution Directory, which allows for more growth and customer participation. See [SAP ICC Integration Scenario - SAP Extension Suite Certification](#) and [SAP Certified Solution Directory](#).

### ⓘ Note

Certification of SaaS solutions is only available if you have used add-on build delivery.

## 4.2.10.1.6.3 Subscribe

Once the SaaS solution has been fully deployed and commercialized, customers can order the offered product. After the buying process has been completed, you, as a SaaS solution operator, are responsible for onboarding the customer.

### 4.2.10.1.6.3.1 Create Consumer Subaccount

Since your solution is only available within your global accounts for production, you as the solution operator must create a consumer subaccount for your customer. After a customer has ordered the SaaS solution, create a 06 Consume subaccount in the global account for production, which will be used to create the subscription. See [Set Up a Global Account for Production \[page 1354\]](#).

While creating the subaccount, you must provide the following details:

- Name: Give the subaccount a unique display name. This can be the name of the customer, a customer number or a specific naming pattern.
- Provider: Choose the IaaS provider according to the IaaS provider selected for the provider subaccount.
- Region: Choose the subaccount region according to the region where the provider subaccount was created. See [Regions and API Endpoints for the ABAP Environment](#).
- Subdomain: The subdomain plays an important role, as it becomes the consumer-specific part of the tenant-specific application URL.

### 4.2.10.1.6.3.2 Subscribe to SaaS Solution

After you as the solution operator have created the consumer subaccount, the subscription to the SaaS solution is enabled for the consumer.

In the consumer subaccount, navigate to the Service Marketplace. To subscribe to the SaaS solution, select the solution and the desired service plan and choose Create. The subscribed application is now displayed in Services Instances and Subscriptions.

The subscription process triggers the provisioning of:

- a new ABAP system in case all other systems have reached the consumer tenant limit.
- a new consumer tenant that is represented by an additional client in the ABAP system.

In both cases, a confirmation mail will be sent to the initial administrator maintained in the solution configuration. You can also monitor the provisioning using the [Systems Overview](#) and [Operations Dashboard](#) apps in the Landscape Portal.

The business type of the consumer tenant that is created depends on the usage parameter configured for your solution. If you have set `usage = prod`, a tenant of business type [Partner Customer Production Tenant](#) is created. If you have set `usage = test`, a tenant of business type [Partner Customer Test Tenant](#) is created. For more information, see [Tenant Business Types \[page 1409\]](#).

#### Note

gCTS Delivery: In case of delivery via gCTS, you, as a SaaS solution operator, can use a separate test subscription that is not related to a customer to trigger the creation of the system before the actual customer subscription. This initial system creation and import of the software components needs to be performed for each customer production system.

### 4.2.10.1.6.3.3 Configure Consumer Subaccount

Depending on the scope of the SaaS solution and customer requirements, you can make the following configurations in the consumer subaccount:

- Trust Configuration: You can configure a custom identity provider for the authentication of subscribed applications in the consumer subaccount. See [Trust and Federation with Identity Providers](#). To restrict access based on certain criteria, such as the IP address, use the SAP Cloud Identity Services - Identity Authentication.

#### Note

With new subaccounts created after September 24th 2020, [automatic creation of shadow users](#) is switched off by default for the default identity provider, SAP ID service. If the creation of shadow users is disabled, any additional business users need to be added manually to the users of the consumer subaccount so that authentication on subaccount level is possible. Business users can only log on to the ABAP tenant if there is an existing employee/business user in the ABAP system.

- Cloud Connector: Using Cloud Connector, you can create a link between SAP BTP applications and on-premise systems. See [Managing Subaccounts](#) on how to add and connect a consumer subaccount to Cloud Connector.
- Destination: Configure destinations, including technical information about the target resource, to connect your application to a remote service or system. See [Managing Destinations](#).

These configurations can be done directly by you, the SaaS solution operator, or by the customer. In the latter case you provide the necessary authorizations by assigning the role collection Subaccount Administrator to a consumer subaccount administrator. The consumer subaccount administrator can then access the consumer subaccount in the SAP BTP cockpit including subscriptions, trust configuration, Cloud Connectors and destinations without having access to other consumer subaccounts in the global account for production.

The consumer subaccount administrator can assign role collection Cloud Connector Administrator to a user to operate the data transmission tunnels used by the Cloud Connector. While adding the consumer subaccount in Cloud Connector, the same Cloud Connector administrator user is being configured.

See [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[Feature Set B\]](#) for more information on available role collections in the consumer subaccount.

#### 4.2.10.1.6.3.4 Initial User Onboarding

In order to be able to start using the solution, your customer needs to receive an initial administrator user in their consumer tenant. This is done automatically via the initial tenant onboarding of your solution.

To trigger this onboarding process, you as the provider need to be assigned to the SolutionAdmin role collection in the consumer subaccount:

1. Log on to the consumer subaccount and navigate to Security Role Collections.
2. Select the role collection SolutionAdmin, choose Edit, scroll down to Users, and select +. Enter the email address of your user, select SAP ID Service as your identity provider or use your preferred custom identity provider, and save your changes.

Now you are ready to access the solution's new tenant, where you will see the initial onboarding screen. In the confirmation dialog, you must confirm details such as email, subdomain, and subaccount ID to onboard an initial user for the customer:

1. Open the URL your provider has sent you to get access to their SaaS application.
2. On the initial administrator onboarding screen, your email address, subdomain, and subaccount ID are displayed. Select Onboard User to start the onboarding process. This might take a few minutes. Once the process is finished, you are redirected to the system. This onboarding process only needs to be performed once.

You can also monitor the tenant user provisioning using the Systems Overview and Operations Dashboard apps in the Landscape Portal. Ongoing tenant user provisionings are displayed in the Requests section of the Systems Overview app. See [Systems Overview](#).

#### 4.2.10.1.7 Configure and Implement a Customer Project

After you have set up the consumer subaccount, tenant, and tenant user, a remaining customer implementation project in the consumer tenant can start. This includes the setup of identity and authentication management, integration with other systems or services, and adjustment of business configuration.

Once you have configured and implemented the customer project, the SaaS solution matches the requirements of the customer and you can provide the SaaS application URL to the customer.

##### → Tip

We recommend providing documentation for your solution that includes specific configuration steps, for example to describe which business roles users have to create. In the following, we only describe the generic configuration steps.

### Note

If you use gCTS for delivery to a customer production system AMT instead of add-ons, the system is initially created without application content. Therefore, you have to clone the required software components manually.

For the initial import of a software component, as a SaaS solution operator on the provider side, you have to clone the software component in the customer production system AMT. The system can be accessed via the *Landscape Portal* application. In the *Landscape Portal*, the *Systems Overview* app provides an overview about the provisioned systems. Open the system that has been created during subscription to the multitenant application. In the *Tenants Overview*, the production tenant (client 100) can then be opened for provider access.

In the *Manage Software Components* app, select a software component and clone the branch of the software component with the product version, for example v1.0.0.

See [Delivery via Add-On or gCTS \[page 1342\]](#).

## Prerequisites

To start with the customer project, you have to know the requirements of the customer, and need to have a consumer tenant subscription and a user for the initial consumer access. See [Subscribe to Multitenant Applications Using the Cockpit](#) and [Initial User Onboarding \[page 1395\]](#).

As the consumer subaccount administrator, access the SAP Fiori launchpad of the SaaS Solution subscription from the consumer subaccount account by navigating to  [Services](#)  [Instances and Subscriptions](#) .

## Set Up IAM

### Create and Assign Business Roles

Before creating and assigning business roles, you have to make sure that business users are already available. They can be created manually or automatically. See [User Provisioning \[page 803\]](#).

As a business user, you configure the authorization to access the applications that are part of the SaaS solution via business roles. With the *Maintain Business Roles* app, you define business roles by combining predefined business catalogs and, if necessary, define value help, read and write access by maintaining the allowed values for fields.

Instead of creating a business role from scratch, you can also use business role templates. See [How to Create a Business Role from a Template \[page 2668\]](#).

### Create Spaces and Pages for Business Roles

The spaces mode offers more flexibility to adjust the SAP Fiori launchpad layout for specific user groups.

As a business user, to create launchpad spaces and pages, see [How to Create Spaces and Pages for a Business Role \[page 2680\]](#).

## Integrate

### Create Communication Arrangements

A communication arrangement is a runtime description of a specific communication scenario created by you as a business user. It describes which communication partners communicate with each other in the scenario and how they communicate. See [How to Create a Communication Arrangement \[page 2596\]](#).

#### Note

Depending on whether you want to use an authentication method for outbound communication that requires a business user context (OAuth 2.0 SAML Bearer Assertion, OAuth 2.0 User Token Exchange, JWT Principal Propagation), you need to configure a destination for communication arrangements in the system instead of maintaining credentials by using an outbound communication user. See [Supported Protocols and Authentication Methods](#) and [Create a Destination](#).

See [Integrating On-Premise Systems \[page 1184\]](#) on how to integrate the SaaS solution with on-premise systems.

## Configure

### Create Business Configuration

As a business user, you can use the [Custom Business Configurations](#) app to adjust business configuration objects to change and influence the system behavior.

See [Custom Business Configurations App \[page 2574\]](#) on how to adjust configuration objects provided in the SaaS solution.

### Configure Key User Extensibility

Key user extensibility that is enabled in the SaaS solution can be configured and consumed in consumer tenants.

#### Note

Key user extensibility provided in a SaaS solution can only be configured in tenants of particular types, for consumption purposes in customer systems, such as AMT, in tenants of type [Partner Customer Test](#) or [Partner Customer Production](#) depending on parameter usage in the configuration of the ABAP solution. See [ABAP Solution Service \[page 1379\]](#).

These tenant types are provisioned in non-development systems, such as customer system AMT, where development is not allowed (`is_development_allowed = false`). The tenants are created dependent on a subscription to the SaaS solution. See [Subscribe to Multitenant Applications Using the Cockpit](#).

As a business user in a [Partner Customer Test](#) or [Partner Customer Production](#) tenant (client  $\geq 200$ ), you configure key user extensibility in a customer production system AMT.

- See [Custom Logic \(Deprecated\) \[page 2633\]](#) for guidance on how to use the [Custom Logic](#) app to create and maintain custom logic for business add-ins (BAAddIs).
- See [Configuring Predefined Custom Fields \[page 2639\]](#) for guidance on how to configure predefined custom fields to customize applications and their UIs.

- See [Adapting SAP Fiori UIs at Runtime - Key User Adaptation](#) for guidance on how to adapt the user interface of an app separately for specific user roles.

## 4.2.10.1.8 Maintain, Monitor, Support

After the go-live of the SaaS solution and the deployment of its updates, consumers using the solution might run into issues that you have to troubleshoot and debug.

### 4.2.10.1.8.1 Maintain

After the initial add-on release has been shipped as a SaaS solution offering, the development of maintenance patches, support packages, and consequent add-on releases comes into play.

Once everything is implemented, built, and the maintenance delivery is deployed, the corresponding changes become available in the customer production system AMT.

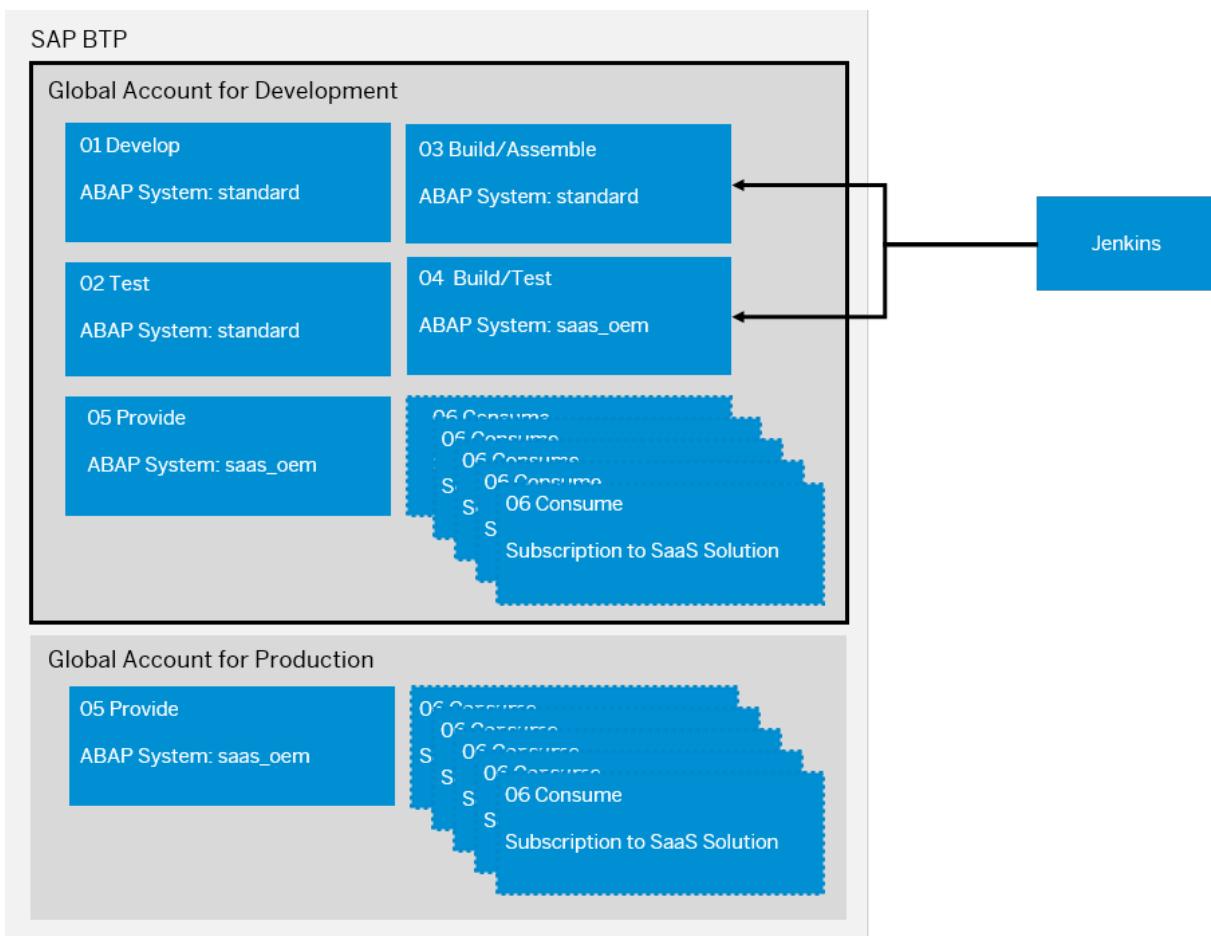
#### Note

New features are developed on different code lines (branches). If you create a so-called maintenance branch to implement patches while new features are implemented in the main branch of a software component, you can implement new features and provide bug fixes at the same time. For more information, see [Versioning and Branches](#).

## Prerequisites

- To set up the maintenance system landscape, you need the relevant entitlements in the global account for development. See [Entitlements and Quotas](#).
- When using a maintenance system landscape, you require business users with authorization to use the Manage Software Components app and developer users using ABAP Development Tools must be available in the systems. See [Manage Software Components \[page 2805\]](#) and [Getting Started as a Developer in the ABAP Environment \[page 217\]](#).
- To configure new add-on versions, you need the existing pipeline configuration for an add-on build pipeline. These can be the pipeline templates in the Build Product Version app or a manual configuration. See [Build Product Version](#) and [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).
- If you have configured the add-on build pipeline manually, you need a Jenkins server where you can execute it.
- To use the Landscape Portal, you need a subscription to the Landscape Portal application and a user assigned to role collection `LandscapePortalAdminRoleCollection`.

## 4.2.10.1.8.1.1 Set Up Maintenance System Landscape



Bug fixes aren't implemented as part of the development code line, but in a separate correction code line. In this manner, corrections can be delivered to consumers even while regular development is ongoing.

We recommend setting up a maintenance system landscape that runs in parallel to the development system landscape. This maintenance landscape consists of:

- A correction system COR where corrections are developed. This system is provisioned in subaccount **01 Develop** in the development space. Set parameter `is_development_allowed = true`.
- A quality assurance system QAS for testing the corrections. This system is provisioned in subaccount **02 Test** in the test space. Set parameter `is_development_allowed = false`.

The required entitlements and system-related details are listed in the table below:

Global Account	Subaccount	Space	Entitlements	ABAP System
Global Account for Development	01 Develop	Develop	1x abap/standard	COR
			abap/hana_comPUTE_UNIT (standard: 2)	
			abap/abap_comPUTE_UNIT (standard: 1)	
	02 Test	Test	1x abap/standard	QAS
			abap/hana_comPUTE_UNIT (standard: 2)	
			abap/abap_comPUTE_UNIT (standard: 1)	

The Manage System Hibernation app in the Landscape Portal allows you to shut down the maintenance systems when no corrections are being developed, significantly reducing the costs incurred. See [Manage System Hibernation](#). For more information on working with two codelines, see [Use Case 2: One Development and Correction Codeline in a 5-System Landscape](#).

It is not strictly necessary to set up a separate maintenance system landscape. You can also implement corrections in the existing development and test systems to further reduce costs. In this case, ongoing development must be stopped while the release branch is checked out and worked on. See [Use Case 1: One Codeline in a 3-System Landscape](#).

## 4.2.10.1.8.1.2 Create Add-On Update

As an add-on administrator, you can provide different kinds of updates. The decision regarding which one to use depends on the motivation behind a given update.

- Small and urgent corrections are delivered as patches
- Collections of less urgent corrections and other functional enhancements should be delivered as support packages
- New release versions are released with updates containing significant enhancements and new features

### → Tip

Use the [Check Product Version](#) app in Landscape Portal to find out which product version has already been built.

#### Create new patch version (emergency patch)

Patch versions are used to deliver unplanned and most likely urgent corrections that are required to keep the application up and running. These changes could be required for a service consumption model of an OData Client Proxy in case of changes to the structure of the underlying OData service. See [OData Client Proxies \[page 901\]](#).

Bug fixes are delivered as a new patch version of a software component and are implemented on the correction code line. Therefore, the dedicated ABAP correction system COR is used. For more information on the different software component version numbers, see [Software Component Version](#).

### Note

While creating patch versions for the same support package level or release version, the same maintenance branch should be used.

To deliver a new patch version, see the following steps:

- **Develop**
  - As an add-on admin, check out the maintenance branch, which is the branch that is created for bug fixes and maintenance deliveries. See [How to Work with Branches \[page 282\]](#).
  - Implement the bug fix in the ABAP correction system  
As a developer, implement the required changes to the software component in the ABAP correction system COR. These changes shouldn't introduce new features because patch versions should only contain small emergency corrections. Release the transport tasks and requests.
- **Test**
  - Import the maintenance branch in the ABAP quality assurance system  
You need to test the implemented corrections by importing the maintenance branch for the current support package level, where the transports for necessary bug fixes were released, in the dedicated test system, which is the ABAP quality assurance system QAS. See [How to Pull Software Components \[page 2810\]](#).
  - Test bugfix in ABAP quality assurance system  
As a tester, test the corrections in the QAS system to validate that the provided bug fixes are solving the problem and are working properly. Required testing can involve anything from maintaining business roles, creating communication arrangements to more complex testing scenarios where connectivity is involved.
- **Double maintenance of corrections into development**  
As a developer, you have to maintain the corrections that have been developed in the ABAP correction system COR as separate transports in development system DEV (so called double-maintenance). See [Double Maintenance of Corrections into Development \[page 759\]](#).
- **Configure new product version**  
As an add-on admin, you can use the Build Product Version app in the Landscape Portal to create new patches:
  - Make sure that the pipeline template for Patch Delivery is configured. If you want to validate the add-on build beforehand, configure the template for Test Patch Delivery as well. See [Configure Pipeline Template](#).
  - Choose the parent version that shall be patched and create a new product version of type Patch Delivery. See [Create a New Product Version](#).

If you are configuring the add-on build pipeline manually, adjust your add-on descriptor file to build the new patch version. See [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).

#### **Create new support package version (hotfix collection)**

Support package versions are used to deliver planned functional enhancements outside of new major releases. They are often used to bundle multiple patch versions to a hotfix collection.

A support package version could be used for example to deliver a new simple business configuration UI. See [Business Configuration in SAP BTP ABAP Environment \(1\): Overview and BC Maintenance Apps](#).

## ⓘ Note

Support package deliveries aren't limited to changes to only one software component. But only software component support packages and patches should be included in a support package delivery. The software components that are part of an add-on product can only be changed as part of a release delivery.

## ⓘ Note

If you use a quarterly delivery approach, new features are not delivered using support package versions. Instead, a new maintenance branch is created from the previous maintenance branch, using systems COR and QAS instead. See [Delivery Models](#).

To deliver a new support package stack, see the following steps:

- **Develop**

- As a developer, implement the new feature in the main branch in the ABAP development system DEV. Bug fixes should be delivered via patch deliveries if they need to be available before the next support package delivery cycle. See [The Add-On Product](#).

- **Test**

As an add-on admin, use the [Manage Software Components](#) app to import the changes in the main branch into test system TST.

- As a tester, test the new implemented features in the TST system to validate the functionality. Required testing can involve anything from maintaining business roles, creating communication arrangements to more complex testing scenarios where connectivity is involved.

- **Create maintenance branches**

As an add-on admin, create a new maintenance branch for each new support package level of a software component. These maintenance branches are used for developing bug fixes and maintaining deliveries. This assures that these bug fixes can be implemented in the correction code line to not block ongoing development on the main branch.

### Create new product version

As an add-on admin, you can use the Build Product Version app in the Landscape Portal to create new support package stacks:

1. Make sure that the pipeline template for Support Package Stack is configured. If you want to validate the add-on build beforehand, configure the template for Test Support Package Stack as well. See [Configure a Pipeline Template](#).
2. Choose the parent version that shall be patched and create a new product version of type Patch Delivery. See [Create a New Product Version](#).

If you are configuring the add-on build pipeline manually, adjust your add-on descriptor file to build the new patch version. See [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).

### Create new release version (product version)

Release versions are used to deliver new major, planned functional enhancements. Typically, they include multiple new implemented features. For example, multiple new apps could be introduced with such a major release.

## ⓘ Note

Only with new release deliveries, you can change the software component version bundle in a new release.

To deliver a new release version, see the following steps:

- **Develop**

As a developer, implement the new feature in the main branch in the ABAP development system DEV. Bug fixes should be delivered via patches or support packages instead. Release the transport tasks and requests.

- **Test**

As an add-on admin, use the *Manage Software Components* app to import the main branch into the test system TST so that new developments can be tested.

As a tester, test the new implemented features in system TST to validate the functionality.

Required testing can involve anything from maintaining business roles, creating communication arrangements to more complex testing scenarios where connectivity is involved.

- **Create maintenance branches**

As an add-on admin, create a new maintenance branch for each new release version of a software component. These maintenance branches are used for developing bug fixes and maintaining deliveries. This assures that these bug fixes can be implemented in the correction code line to not block ongoing development on the main branch.

- **Configure new product version**

As an add-on admin, you can use the Build Product Version app in the Landscape Portal to create new release versions:

- Make sure that the pipeline template for Release Delivery is configured. If you want to validate the add-on build beforehand, configure the template for Test Release Delivery as well. See [Configure Pipeline Template](#).
- Choose the relevant add-on product and create a new product version of type Release Delivery. See [Create a New Product Version](#).

If you are configuring the add-on build pipeline manually, adjust your add-on descriptor file to build the new patch version. See [Build and Publish Add-on Products on SAP BTP, ABAP Environment](#).

### gCTS Delivery

If you use gCTS for delivery, the process of creating an update for SaaS solutions is grouped into urgent corrections (patch version) and new releases (release version). See [Use Case 2: One Development and Correction Codeline in a 5-System Landscape](#) and [Delivery via Add-On or gCTS](#).

For delivering an emergency patch via gCTS, see section Urgent Corrections in [Use Case 2: One Development and Correction Codeline in a 5-System Landscape](#)

For delivering a new major version via gCTS instead of using add-ons, see section For Go Live/Development after Go Live (Including Deferrable Corrections) in [Use Case 2: One Development and Correction Codeline in a 5-System Landscape](#).

For general information regarding gCTS delivery, see [Delivery via Add-On or gCTS](#).

## 4.2.10.1.8.1.3 Trigger Add-On Build Pipeline



Similar to the build of the initial add-on version, as an add-on administrator, you need to trigger the execution of the configured ABAP environment pipeline for an add-on build. You can do this using the Build Product Version button in the corresponding app. After a successful build, the new add-on version is technically available for deployment to the ABAP environment. See [Create a New Product Version](#). If you are configuring the add-on build pipeline manually, then you need to trigger its execution in your Jenkins server. For in-depth information about the ABAP environment pipeline, check out [ABAP Environment Pipeline](#).

#### *gCTS Delivery*

If you are using gCTS instead of add-ons for delivering software components into production systems, an add-on build is not required.

##### **ⓘ Note**

Please ensure that the add-on product version to be published is properly tested before confirming the release decision. This includes testing in SAP Fiori launchpad and the ABAP Test Cockpit. See [Test in the ABAP Environment SAP Fiori Launchpad \[page 1365\]](#) and [Test in the ABAP Test Cockpit \[page 1366\]](#).

## **4.2.10.1.8.1.4 Check Add-on Build Result**

Use the [Check Product Version](#) app in Landscape Portal to check whether the product version, its components, and respective packages are ready for delivery. See [Check Product Version](#).

## **4.2.10.1.8.1.5 Deploy Add-On Update**

As a SaaS solution operator, you can apply add-on updates to existing systems via the [Deploy Product](#) app in the Landscape Portal.

You can select the target add-on product and version for the update. Afterwards, you can select the systems where the update should be applied and schedule the update.

##### **ⓘ Note**

If you are using gCTS instead of add-ons for delivering software components into production systems, you need to pull the changes of all relevant software components into all productive systems using the [Manage Software Components](#) app. An update deployment via the Landscape Portal is not required.

## **4.2.10.1.8.2 Manage Incidents**

### **Incident Management System**

As a provider of a SaaS solution, you should establish an incident management system, such as SAP Resolve, so that consumers can report issues.

If consumers of your SaaS solution run into problems, they need to be able to contact the provider of the software in a standardized way.

#### Information to Identify ABAP System and Consumer Tenant for Incident Processing

To process an incident, you have to identify the ABAP system and tenant, where the consumer issue has occurred. Therefore, the following information is required:

- SAP Fiori launchpad URL  
In the SAP Fiori launchpad user actions menu, the SAP Fiori launchpad URL of a consumer tenant is displayed as *Server* in the [Settings](#) dialog. See [Settings](#).
- System Details  
In the user actions menu, information about the ABAP system and tenant is displayed in section *System* in the [About](#) dialog. See [About](#).  
The system name and description are of particular interest.

#### Information to Reproduce the Customer Issue

To reproduce the consumer issue, the following information should be provided:

- Application details  
In the user actions menu, information about the SAP Fiori application is displayed in the [About](#) dialog in the *Application* section. See [About](#).  
The application title and technical component ID are of particular interest.
- Error details  
The error message is displayed in the SAP Fiori application or as a status code of the HTTP request that is failing.
- Business user details  
In the user actions menu, the name and email address identifying the business user of the consumer are displayed in the [Settings](#) dialog in the *User Account* section. See [Settings](#).
- Step-by-Step Description of the User Actions and Input  
To reproduce the customer issue, you need detailed information about the user actions that lead to the error, and, if possible, user input that was performed during these steps.

### 4.2.10.1.8.3 Troubleshoot and Debug

If a customer faces an issue, you have to provide troubleshooting based on a reported consumer incident. You can access the SAP Fiori launchpad of the consumer tenant or the backend using ABAP Development Tools to further analyze the issue.

#### Identify ABAP System and Consumer Tenant

The ABAP system can be identified in the [Systems Overview](#) in the Landscape Portal application. See [Systems Overview](#).

The system ID is defined in the ABAP Solution service via configuration parameter `sap_system_name`. If there are multiple solutions, the system description identifies the corresponding ABAP system as it always follows the same pattern: *ABAP Solution System for <parameter "name" in ABAP Solution service>*. See [ABAP Solution Service \[page 1379\]](#).

Based on the SAP Fiori launchpad URL of a consumer tenant, the consumer subaccount subdomain, that uniquely identifies the tenant, can be derived. The pattern to extract the subdomain is defined in

`TENANT_HOST_PATTERN`. This is defined as environment variable of the deployed approuter, which is part of the multitenant application. See [Approuter Application \[page 1386\]](#).

### ❖ Example

Identifying the subdomain with a multitenant application created based on the MTA-based approach, see [Multitenant Applications \[page 1376\]](#).

- `TENANT_HOST_PATTERN = (*.*)${route-prefix}.${app-domain}`
- `${route-prefix} = <empty>`
- `${app-domain} = mydomain.com`
- SAP Fiori Launchpad URL = `myconsumer.mydomain.com`

In this example, `myconsumer` is the subdomain identifying the consumer tenant.

The subdomain is displayed in the *Tenants View* as *Subaccount Domain* when selecting the system in the Landscape Portal app and thus can be used to find the consumer tenant (business type *Partner Customer Test* and *Partner Customer Production*).

## Access the Consumer Tenant of the ABAP System

A consumer tenant in client  $\geq 200$  (business type *Partner Customer Test* or *Partner Customer Production*) can be accessed with a provider support user created in the Landscape Portal. See [Create Support Users](#).

- **Frontend Access to SAP Fiori launchpad**

The SAP Fiori launchpad of a consumer tenant in client  $\geq 200$  can be accessed via the link listed in the tenant overview in the Landscape Portal. A provider support user must be requested for the tenant beforehand. Such a user can only access a limited set of SAP-provided apps, for example for Identity and Access Management and Communication Management.

- **Backend Access using ABAP Development Tools**

Access to a consumer tenant in client  $\geq 200$  using ABAP Development Tools requires an adapted service key in the service instance in the SAP BTP cockpit: The service key `SAP_ASP` displayed in the cockpit needs to be adjusted with the GUID of the consumer tenant. See [Create Support Users](#).

ABAP Development Tools access with a provider support user created in the Landscape Portal grants access to troubleshooting capabilities, such as those granted with business role `Application Support Engineer - Development Support (BR_APPL_SUP_ENG_DEV_SUP)`.

## Troubleshoot

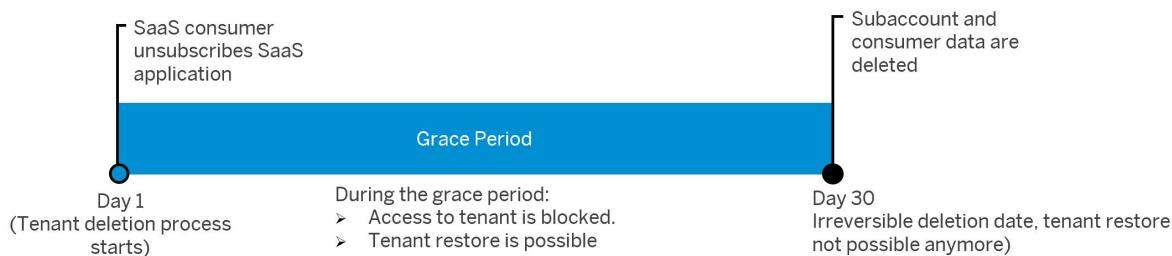
The most common use case for troubleshooting is to debug a business user or communication user executing the implemented business logic that is causing an issue in the consumer tenant. See [Troubleshoot Custom Apps \[page 2988\]](#).

With a provider support user, you need to connect with ABAP Development Tools to the system and consumer tenant, set a breakpoint in the logic, and afterwards the business user of the consumer must reproduce the error.

Apart from debugging, you can also use other troubleshooting tools in ABAP Development Tools to analyze issues in a consumer tenant. See [Troubleshooting Tools \[page 2991\]](#).

#### 4.2.10.1.9 Dismantle

As soon as a consumer unsubscribes from the SaaS solution, an immediate tenant decommissioning process takes place while taking a “grace period” of 30 days into account, according to the following legal obligations:



During the grace period

- The tenant access is blocked for the consumer and for the SaaS provider as well in the support use case.
- The provider can restore the tenant in the *Landscape Portal*. See [Restore Consumer Tenants](#).

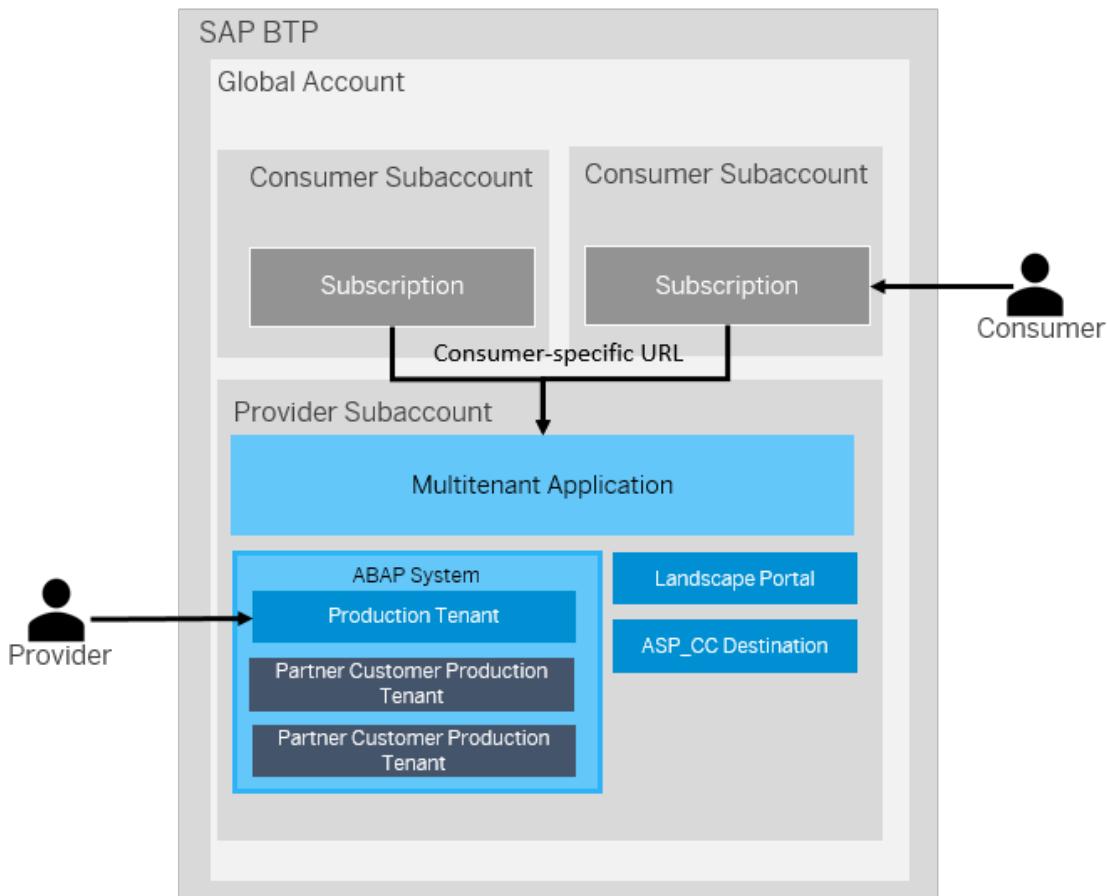
After the end of the grace period, the tenant and all related data are irreversibly deleted and cannot be restored anymore.

Once the last consumer unsubscribes from the SaaS solution, the provider needs to delete the empty system in the BTP cockpit.

Alternatively, as a SaaS solution operator, you can use the *Subscription Management Dashboard* in the provider subaccount to unsubscribe tenants centrally without triggering unsubscription in the consumer subaccount. See [Using the Subscription Management Dashboard](#).

#### 4.2.10.2 Multitenancy in the ABAP Environment

Multitenancy defines the capability to host different customers (tenants) on a single, shared computing infrastructure in order to optimize administration and significantly reduce TCO. A tenant is an organizationally independent unit whose IT business is entirely or partly operated together with the businesses of other tenants by a hosting provider. Multitenancy is especially relevant in a “Software as a Service” (SaaS) business model where customers subscribe to hosted software solutions rather than buying and installing them. For hosting providers, the operation costs per user per tenant are a decisive factor - particularly in the SME market, where typically each tenant only has a small number of users (10 – 50) accessing the hosted solution. Hosting providers can only operate profitably if they succeed in keeping the administration and maintenance costs for the hosted solution as low as possible. This cost pressure also limits the degree of individualization and customization a hosting provider is willing to accept for a hosted solution provided to many tenants.



Multitenancy in the ABAP environment enables independent software vendors or partners (referred to as **application provider**) to develop and operate ABAP solutions as software as a service (SaaS) leveraging SAP BTP infrastructure while hosting several consumers on the same ABAP system. The resources on SAP BTP platform consumed by the solution are paid for by the application provider.

Application **consumers** (= end customers of the provider) subscribe to a provider's multitenant application and use it in a specific consumer subaccount (= tenant). Consumers access the provider's SaaS application via a consumer-specific URL. Consumers cannot see the data of other consumers and Identity and Access Management is kept isolated between different tenants.

The **Multitenant Application** is deployed to the provider subaccount and serves as the entry point for the consumer-specific URLs, so that requests are routed to the corresponding consumer tenant in the ABAP system. Only after the multitenant application has been deployed, the application will be available for subscription to consumers. See [Multitenant Applications \[page 1376\]](#).

The **ABAP system** used to serve the application to the consumers is provisioned in the provider subaccount during the first subscription as abap/standard or abap/saas\_oem system (depending on whether add-on is used for delivery). See [Creating an ABAP System \[page 180\]](#). Different tenants are created as separate clients in the system. Tenants in the ABAP system have different capabilities represented by the tenant business type and a lifecycle status. The ABAP system contains by default a tenant used by the application provider (client 100) for system-level operations like the import of software components to the system. For each subscription to the multitenant application, a tenant used by the consumer (client >= 200) is created. If any consumer tenants still exist in the ABAP system, the system cannot be deleted.

The **Landscape Portal** functions as a central plane for tenant management that allows providers to perform lifecycle management operations such as add-on updates, creating test tenants or support users and more. For more information on how to access and use the Landscape Portal, see [Landscape Portal](#).

The **ASP\_CC Destination** is a destination created on subaccount-level in the provider subaccounts. It points to the Cloud Foundry Cloud Controller API and is utilized by the multitenant application to create the ABAP system in the Cloud Foundry space/org where the multitenant application is deployed. See [Cloud Controller Access Destination \[page 1388\]](#).

#### ⓘ Note

When building tenant-aware applications on top of the ABAP environment, providers need to follow specific ABAP implementation rules to ensure a content separation between different consumers. To view these guidelines, see [Multitenancy Development Guideline \[page 1412\]](#).

### 4.2.10.2.1 Tenant Business Types

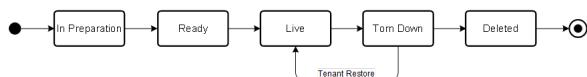
The capabilities of a tenant used by the application provider or application consumer depend on the tenant business type. Business types of tenants that reflect how the tenant is used are derived from the system business type (partner development or production system via parameter `is_development_allowed`).

- **Target Group:** Whether the tenant is primarily used by the application provider or application consumer.
- **ABAP System Client:** Consumer/Provider tenants are created as a different ABAP system client for content separation.
- **Business User Logon:** Business users are authenticated in an ABAP tenant using trusted identity providers configured in either the provider subaccount or consumer subaccount. See [Trust and Federation with Identity Providers](#).
- **Provider Support Access:** In the Landscape Portal the application provider can request support user access for certain tenants in the ABAP system to troubleshoot & debug issues. See [Create Support Users](#).
- **Key-User Extensibility Configuration:** Key-user extensibility features of a solution like custom logic and predefined custom fields can only be configured in tenants of particular types. See [Providing Business Add-Ins \[page 1420\]](#) and [Configuring Predefined Custom Fields \[page 1426\]](#).
- **Software Component Import:** The Manage Software Components app is only available in tenants of particular type. See [Manage Software Components \[page 2805\]](#).
- **Tenant Onboarding:** When and how tenants are created differs depending on the business type. See [Subscribe to Multitenant Applications Using the Cockpit](#).
- **Initial Administrator Onboarding:** In each tenant an initial user is created based on an e-mail address to get access to the tenant after it has been created.
- **Tenant Offboarding:** When and how tenants are deleted differs depending on the business type. See [Dismantle \[page 1407\]](#).
- **Tenant Restore:** Only tenants of particular types can be restored during the retention period. See [Restore Consumer Tenants](#).

	Partner Development System		Production System			
	Partner Development Tenant	Partner Test Tenant	Production Tenant	Partner Test	Partner Customer Test Tenant	Partner Customer Production Tenant
<b>Target Group</b>	Application Provider	Application Provider	Application Provider	Application Provider	Application Consumer	Application Consumer
<b>ABAP System Client</b>	100	>=200	100	>=200	>=200	>=200
<b>Business User Logon</b>	Logon via Identity Provider configured in provider subaccount	Logon via Identity Provider configured in provider subaccount	Logon via Identity Provider configured in provider subaccount	Logon via Identity Provider configured in provider subaccount	Logon via Identity Provider configured in consumer subaccount	Logon via Identity Provider configured in consumer subaccount
<b>Provider Support User Access</b>	Not available	Not available	Not available	Not available	Available via Landscape Portal	Available via Landscape Portal
<b>Key-User Extensibility Configuration</b>	Not available	Not available	Not available	Available	Available	Available
<b>Software Component Import</b>	Manage Software Components app available	Not available	Manage Software Components app available	Not available	Not available	Not available
<b>Tenant On-boarding</b>	Created during system provisioning	Test Tenant created in Landscape Portal	Created during system provisioning	Test Tenant created in Landscape Portal	Created during subscription to solution with parameter usage = test	Created during subscription to solution with parameter usage = prod
<b>Initial Administrator On-boarding</b>	Created during system provisioning based on provider_a dmin_email parameter	Created during test tenant creation in Landscape Portal	Created during system provisioning based on provider_a dmin_email parameter	Created during test tenant creation in Landscape Portal	Created using initial user on-boarding form after subscription	Created using initial user on-boarding form after subscription
<b>Tenant Off-boarding</b>	Deleted during system decommissioning	Deleted in Landscape Portal	Deleted during system decommissioning	Deleted in Landscape Portal	Deleted automatically after retention period	Deleted automatically after retention period
<b>Tenant Restore</b>	Not available	Not available	Not available	Not available	Consumer Tenant restore using Landscape Portal	Consumer Tenant restore using Landscape Portal

## 4.2.10.2.2 Tenant Lifecycle

Each consumer tenant goes through a certain lifecycle starting from the provisioning to the deprovisioning of the tenant and possibly tenant restoration. The current lifecycle status of a tenant in the ABAP system is shown in the Landscape Portal.



Request Type in Landscape Portal	Changes to Lifecycle Status	Description
New Tenant	In Preparation	The creation of a new tenant starts with status <b>In Preparation</b> . After the creation of the new tenant in the system, the registration of client and tenant in the system, and the import of initial tenant data, the tenant is <b>Ready</b> . Finally, after preparation of logon to the new tenant, technical configuration, business configuration, activation of tenant features, setup of monitoring, the tenant is finalized, and the lifecycle status is set to <b>Live</b> .
	Ready	
	Live	
Delete Tenant	Live	Tenant deletion is triggered when the consumer subscription is deleted. The deletion request changes the tenant lifecycle status from <b>Live</b> to <b>Torn Down</b> . During a retention period of 30 days, the tenant remains in status <b>Torn Down</b> , reflecting the deletion in progress. After 30 days, and if the tenant is not restored, the tenant deletion is automatically completed, and the status is changed to <b>Deleted</b> .
	Torn Down	
	Deleted	
Restore Tenant	Live	After a tenant deletion was requested by deleting the corresponding consumer subscription, the tenant can be restored in the Landscape Portal during a retention period of 30 days. Doing so, the lifecycle status is changed back from <b>Torn Down</b> during the retention period, to <b>Live</b> after the completed tenant restoration.
	Torn Down	
	Live	

## 4.2.10.2.3 Multitenancy Development Guideline

Multitenancy is required if you want to run several customers on the same ABAP system. When building tenant-aware applications on top of the ABAP environment, you must follow dedicated rules to ensure, for example, a content separation between different customers.

### Architecture Overview and Basic Concept

The following sections explain how to host multiple tenants in one ABAP system by building a multitenant enabled solution. We use multiple clients in the ABAP environment as a technical means to implement multitenancy.

There are the following basic principles regarding multitenancy:

- for the **tenant**: Strict data isolation is guaranteed between tenants on all levels of the system. It must be ensured that the tenants' data as well as their tenant-specific application extensions are under no circumstances mingled with the data or applications of any other tenant being operated in the same system.
- for the **hosting provider**: The costs for operating multiple tenants in the same system must be significantly lower compared to single tenant systems. Multitenant systems must reduce the TCO of a hosted solution in such a way that the business case is met.

#### ⓘ Note

Multitenancy is only one means to reduce the marginal costs of operations addressing system and hardware costs.

To some extent, these principles are contradictory. The first principle could be most easily fulfilled by single tenant systems, whereas the second principle imposes the need for sharing resources on each system level, for example, hardware, database, application server. Multitenancy can thus be described as sharing as much system resources as possible without violating the principle of tenant data isolation.

#### ⓘ Note

Tenant isolation isn't completely ensured by the ABAP environment infrastructure. However, the following sections describe the design principles how multitenancy can be achieved in the ABAP environment by an evolution of the present multiclient capability of an ABAP system.

The ABAP environment is a Platform as a Service (PaaS) offering for ABAP. In a typical scenario, an independent software vendor (ISV) develops and runs a Software as a Service (SaaS) application in the ABAP environment and sells this application to end-customers. In order to avoid confusion with the term "customer", as the ISV is also a customer of the ABAP environment, the following terminology is used in this document:

- The **service provider** of the SaaS application is responsible for developing, maintaining, and operating the application on top of the ABAP environment. The service provider is typically an ISV or a development partner.
- The **provider system** is an ABAP system in the ABAP environment that runs the SaaS application. The provider system is operated by the service provider and offers multitenancy capabilities.
- The **service consumer** of the SaaS application is an end-customer of the service provider who has subscribed a tenant in a provider system.

## Assumptions and Limitations

- Multitenancy is only supported in provider systems operated by the same service provider to run a single SaaS application.
- Multitenancy isn't supported for the development scenario using the ABAP Development Tools (ADT). This means that several service providers can't share the same system for development. Furthermore, a consumer can't further develop or extend the provider's SaaS application using ADT.
- You as a service provider can build a multitenancy application following the respective guidelines.
- Transport of objects isn't supported in a multitenant environment. Business configuration content is usually transported to a productive environment by using the transport API `IF_A4C_BC_HANDLER`, in this case, the user interface doesn't have to be implemented. To transport business configurations between systems, you can use the download/upload functionality, see [Upload Business Configuration \[page 2579\]](#).
- There is no system-controlled process to ensure that you as a service provider build your application in a multitenancy-compliant way. If strict data isolation is a legal requirement, we propose an external audit to ensure that your code doesn't break data isolation.
- There are options to share data between tenants. Furthermore, native SAP HANA database access via ABAP-managed database procedures (AMDP) requires special considerations. It's only documented which functionality does have which impact on client isolation. There is no check ensuring that you as a service provider use these features in a multitenancy-compliant way.
- To ensure tenant lifecycle procedures, the applications must be built in such a way that the procedures for tenant copy, tenant move, and tenant delete can deal with it.

### Note

If you don't comply with these aspects of the guideline, this causes issues where SAP, you as a provider, and the consumer are involved.

- Security and system logs store data in a cross-client persistence together.
- Any multitenancy-specific certifications from SAP in addition to the already existing ABAP environment product certifications are out of scope.

## Guidelines for Multitenancy Applications in the ABAP environment

The following sections cover the relevant guidelines for content separation in multiclient systems. We use the multiclient architecture of the ABAP system for multitenancy enablement and list the design principles to reach multitenancy.

## Prerequisites

- Store tenant-related data in client-dependent tables of type A, C, or L.
- Store system-related data in client-independent tables of type S.
- Always add the selection of the client to ABAP database procedures (AMDPs).

Make sure consumers cannot modify the client parameter or any other part of the AMDP using the application or by tampering requests.

- Don't generate development objects or other client-independent data system-locally in the provider system.
- Don't evaluate the actual value of the 3-digit client field (IF sy-mandt = 'nnn' .... ENDIF).

## Database Table Design

You have to classify database tables according to their content. There are the following types:

- Tenant Content (client-dependent)
  - Tenant configuration data – tables with delivery class "C"
  - Tenant application data – tables with delivery class "A"
  - Tenant temporary data – tables with delivery class "L"

Database tables for tenant content must be client-dependent. This means that the first field of the table must be of datatype "CLNT". We recommend using the inline declaration „abap.clnt“.

Only the content of client-dependent "C" and "A" tables is considered during tenant copy and tenant move. Content of client-independent tables which are not delivered from the development system and "L" tables are lost during tenant lifecycle processes such as tenant move.

During tenant delete, the content of all client-dependent tables is removed.

The delivery class must be "C", "A", or "L".

The delivery classes "E", "G" and "W" are not supported in the ABAP environment at all.

- System Content (client-independent)
  - System configuration data – tables with delivery class "S"

Store data that is defined by the service provider and not specific for any tenant in a client-independent "S" table. Define the content in the respective development system and export it as TABU entries via a development transport request. The content is considered as code and imported like other development artifacts into subsequent systems such as the provider system.

## Database Table Access

- Read Access Using ABAP SQL

Whenever you access table content, ensure that you don't violate tenant isolation. ABAP SQL does not support additions such as CLIENT SPECIFIED or USING CLIENT in the ABAP environment. As a result, access to other clients than SY-MANDT is technically not possible. But you have to define client-dependent tables properly so that the ABAP runtime applies this role accordingly.

- Write Access Using ABAP SQL

Do not write "S" tables in the provider system.

### ⓘ Note

If you write "S" tables in the provider system, this content is not considered during tenant lifecycle management events, such as tenant move, and you lose content. Furthermore, you violate tenant isolation, since every tenant has immediate access to the changes.

- Access Using AMDP

Access to SAP HANA database using AMDP is not controlled via the ABAP database interface. The client is just handed over as a parameter. This can easily violate the content separation and access a table data for another client. Make sure that the client is always passed via SY-MANDT.

Consumers must never be able to modify the client parameter or any other part of the AMDP using the application or by tampering requests.

### ⓘ Note

In the future, it is highly recommended to use the AMDP option CDS SESSION CLIENT DEPENDENT or CLIENT INDEPENDENT.

## Code Generation

Development artifacts such as Data Dictionary objects, classes, or interfaces, which are typically created via ABAP Development Tools (ADT), are always client-independent. Changes are always recorded in transport requests.

### ⓘ Note

It is not allowed to generate development artifacts in a non-development system, such as a provider system. This generation violates the content separation and is not considered during tenant move and copy procedures. You are only allowed to generate development artifacts in **development systems** in the ABAP environment. You can use the XCO library for that purpose. If you generate development artifacts, the service provider is responsible for ensuring that the generated artifacts are transported into all required provider systems. Standard lifecycle events such as a tenant move don't consider such client-independent artifacts.

## Multitenant-Aware Programming

- Cross-Client Locking (Enqueue)

Lock Objects are associated to database tables or structures. The runtime API in the ABAP environment is `CL_ABAP_LOCK_OBJECT_FACTORY` and the returned instance of type `IF_ABAP_LOCK_OBJECT`.

### ⓘ Note

Only lock client-dependent database tables or structures having a `CLIENT` field.

- `CL_ABAP_CONTEXT_INFO=>GET_SYSTEM_DATE` and `TIME`

The system date/time (methods `CL_ABAP_CONTEXT_INFO=>GET_SYSTEM_DATE`, `CL_ABAP_CONTEXT_INFO=>GET_SYSTEM_TIME`) is always based on UTC time zone. The system time

zone can't be changed tenant-specifically. As a usual approach in cloud environments, it needs to be considered in your code, as it gives the same date and time for all tenants.

- Usage of System Information (sy-sysid and sy-mandt)

Storing SY-SYSID and SY-MANDT explicitly in database tables is not allowed. This would lead to issues after tenant copy or tenant move, because both values are not unique and not stable in the system landscape. A dedicated API is provided to get a unique and stable identifier for the tenant. See [Tenant \[page 1806\]](#).

## Connectivity

Outbound communication management in the ABAP environment differs from the on-premise usage of SM59 destinations. See [Developing External Service Consumption \(Outbound Communication\) \[page 890\]](#). Destinations are configured in a Cloud Foundry destination service and the ABAP consumer API references these destinations.

### ⚠ Restriction

The ABAP environment only supports one communication configuration layer that is intended for the consumer. It is currently not possible to enforce a communication configuration exclusively managed by the service provider within the consumer tenant.

With regard to inbound communication, inbound RFC is not entirely supported.

## Limitations

There might be some use cases where the guidelines provided are not applicable, for example, when sharing data across tenants or sharing aggregated customer data.

For such use cases, the operations concept needs to be documented and an alignment with SAP is required. We recommend checking the operational impact on tenant move and tenant copy procedures. In addition, we recommend a review for multiclient content separation.

## Usage of SAP-Delivered SAP Fiori Apps

SAP has delivered several SAP Fiori apps for the administration of an ABAP environment system. These SAP Fiori apps are targeting different roles, such as administrator, project manager, and developer. Regarding multitenancy, some of these apps are covering client-independent objects and configuration according to their purpose. For example, the SAP Fiori app Manage Software Components allows you to import software components, which has an immediate impact on all tenants. Therefore, you need to carefully choose which SAP-delivered business catalogs to include in the business roles of your SaaS application.

## 4.2.10.2.4 Troubleshooting

Find out which steps you can take to identify and resolve specific issues that may come up when developing a multitenant application.

### SaaS subscription error

- The [Landscape Portal](#) subscription might be missing. Make sure the [Landscape Portal](#) is subscribed to in the provider subaccount. This needs to be done before the SaaS subscription.
- The ASP\_CC (CF Cloud Controller) destination might have failed: An ASP\_CC destination with the correct credentials/URLs must exist in the provider subaccount (see [Cloud Controller Access Destination \[page 1388\]](#)). You can use a connection check in the destination for a basic connectivity test. You can also test if the user that was used in the ASP\_CC destination is able to authenticate with the following cf command:  
`cf login -a <cf-api-endpoint e.g. https://api.cf.eu10.hana.ondemand.com> -u <destination-user-name> -p <password> -s <provider-space> -o <provider-org>`. Make sure the following is the case:
  - You have the credentials you require (space developer user). This user is a CF Platform User, i.e. it is either an S-User or a P-User, usually created using a distribution list so that it is a non-personalized e-mail address. A technical communication user that can also be created similar to normal S-Users cannot be used here.
  - You are using the correct URL: the URL needs to be the same as the CF API URL in the provider subaccount, depending on the region, see [Region](#).
  - You are using the right Token Service URL for the region.
  - You are using a technical user as recommended, not a personal user. Personal user accounts can cause issues due to password changes etc.
- The space developer user might be missing: The user used in the credentials of the ASP\_CC destination must be added as space developer in the space of the multitenant application.
- There might be missing entitlements for ABAP systems: During the subscription, the multitenant application/ASP automatically provisions ABAP systems. Make sure that the necessary entitlements are added to the provider subaccount:
  - abap/standard (for delivery via gCTS) or abap/saas\_oem (for add-on based delivery), depending on the configuration of ASP parameter addon\_product\_name. See [Multitenancy in the ABAP Environment \[page 1407\]](#) and [ABAP Solution Service \[page 1379\]](#).
  - abap\_compute\_unit and hana\_compute\_unit depending on the configuration of ASP parameter size\_of\_runtime and size\_of\_persistence. See [Multitenancy in the ABAP Environment \[page 1407\]](#) and [ABAP Solution Service \[page 1379\]](#).
- There might be missing entitlements for multitenant app deployment. Make sure that the necessary entitlements are added to the provider subaccount:
  - Cloud Foundry Runtime entitlement for the approuter in the multitenant app.
  - ABAP Solution Service.
- In case of error "Illegal given domain name" during subscription:
  - The subscription fails because in the saas-registry service instance the URL defined for onSubscription and getDependencies callbacks does not match the pattern of the cis-... route assigned to the approuter application. Please make sure that the route defined in your saas-registry configuration for callbacks has the same pattern as defined in the cis-... route.

## Cannot delete ABAP service instance

- If you cannot delete an ABAP service instance, the ABAP service instance might still contain *Partner Test*, *Partner Customer Test*, or *Partner Customer Production* tenants. Make sure you delete these tenants first by unsubscribing from them (for tenants created via consumer subscription) or deleting the tenants in the *Landscape Portal* (for test tenants created directly in the *Landscape Portal*).
- The ABAP service instance might still contain a SAP\_ASP service key: The service key needs to be deleted before the service instance can be deleted.

## Multitenant application cannot be undeployed

- Always use MTA Undeployment (cf undeploy <MTA ID> --delete-services) instead of deleting service instances individually.
- Check if the saas-registry service instance still contains active subscriptions: the saas-registry instance can only be deleted once there are no more active subscriptions. Unsubscribe using the [Subscription Management Dashboard](#).

## Cannot create consumer-specific route according to `TENANT_HOST_PATTERN` (in the development phase)

### ↔ Sample Code

```
TENANT_HOST_PATTERN: (*.*)-my-consumer-subdomain-productabcdefgijklmnopq-saas-solution-dev.cfapps.eu10.hana.ondemand.com
Consumer Subaccount Subdomain: my-consumer-subdomain
Route Domain: cfapps.eu10.hana.ondemand.com
Route Hostname: my-consumer-subdomain-productabcdefgijklmnopq-saas-solution-dev (64 characters)
```

- Keep in mind that the hostname of the route in CF is limited to 63 characters.
- In the development phase, the hostname part of the route should consist of the consumer subdomain, separator (“-”), and the appName.
- Either shorten the subdomain of the consumer subaccount or shorten the appName used in `TENANT_HOST_PATTERN`.

## After deployment of MTA for multitenant application, no abap system is created

- ABAP systems are automatically created by the ABAP Solution Provider (ASP) during the first subscription.

## Issues with parameter consumer\_tenant\_limit

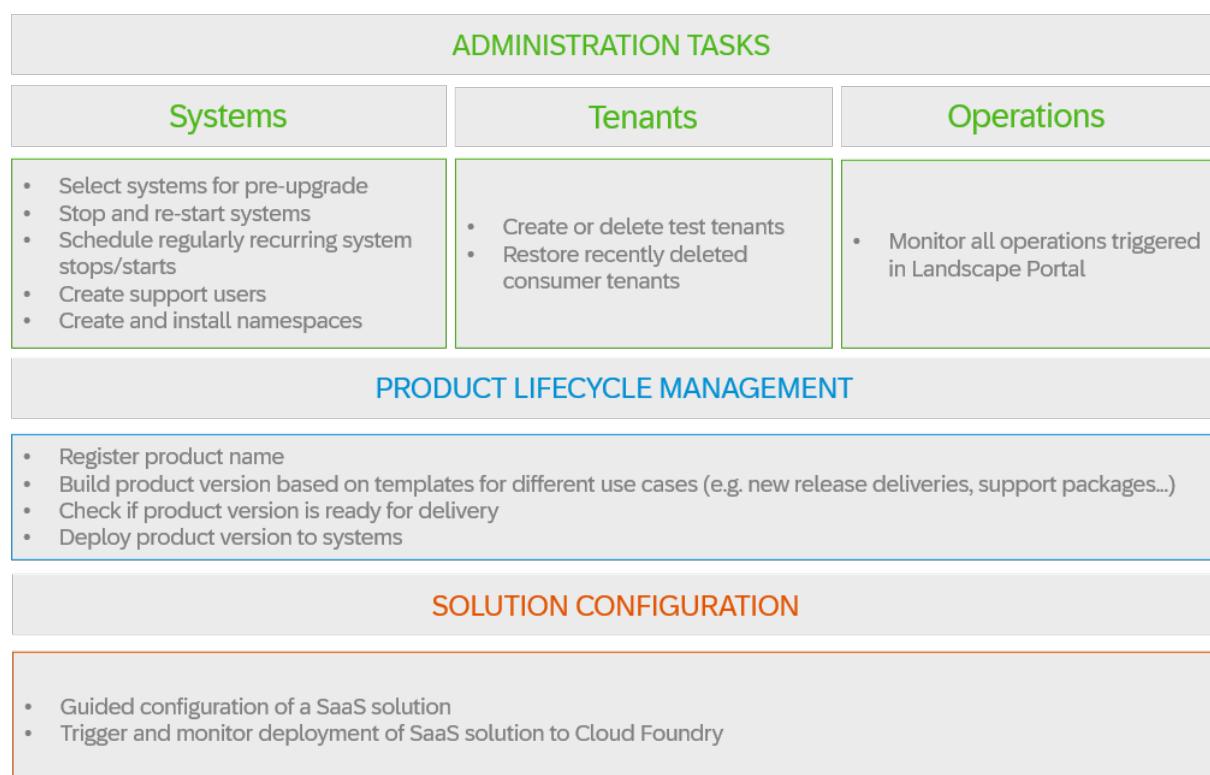
- Keep in mind that recently deleted consumer tenants are counted towards the tenant limit. Systems are automatically created.

## First consumer tenant is missing application content after subscription

- Only in case of delivery via add-on, the application content is installed before the consumer tenant is accessible. In case of delivery via gCTS, the software components need to be imported manually using the [Manage Software Components](#) app in client 100 of the system.

### 4.2.10.3 Landscape Portal

The [Landscape Portal](#) acts as a central tool to allow service providers to perform administration tasks such as hibernating systems, creating test tenants and support users, selecting systems for a pre-upgrade, but also lifecycle management operations such as building, deploying and updating product versions, and even guides you through the configuration and deployment of multitenant SaaS solutions.



For more information about how to access and use the Landscape Portal, see [Landscape Portal](#).

## 4.2.10.4 Extending SaaS Applications

Learn more about extending SaaS applications.

### [Providing Business Add-Ins \[page 1420\]](#)

You, as an SAP partner, want to provide your customers with business add-ins (hereafter called BAdls) so that customers can add their own business logic to your business processes by using the *Custom Logic* app.

### [Configuring Predefined Custom Fields \[page 1426\]](#)

### 4.2.10.4.1 Providing Business Add-Ins

You, as an SAP partner, want to provide your customers with business add-ins (hereafter called BAdls) so that customers can add their own business logic to your business processes by using the *Custom Logic* app.

This document describes the steps required for releasing a BAdl that can be used by key users of the *Custom Logic* app. The following steps are required:

- Create an enhancement spot for your BAdls.
- Create the BAdl definition. This includes the interface definition and filter definitions.
- Release the BAdl definition.
- Provide an example implementation for your BAdl.
- Register the BAdl definition for the *Custom Logic* app.

After following these steps, your customers will be able to create BAdl implementations using the *Custom Logic* app.

#### 4.2.10.4.1.1 Creating an Enhancement Spot for BAdls

##### Context

Creating an enhancement spot for your BAdls allows you to group BAdl definitions that you want to register for the *Custom Logic* app together in one place.

##### Procedure

Create a BAdl enhancement spot in ABAP Development Tools (ADT) using the creation wizard, as described in [Creating BAdl Enhancement Spots](#).

## 4.2.10.4.1.2 Creating the BAdl Definition

Create a BAdl definition to provide your business add-in to the customer.

When creating a BAdl definition, please pay attention that some features may not be used, as indicated in the table below. When registering your BAdl for use within the *Custom Logic* app, make sure to perform a compatibility check to verify that all restrictions are met.

BAdl Option Restrictions

Option	Description	Restriction	Comment
Description	The short description of the BAdl definition	Mandatory	Please keep the short description clear and intuitive for frontend display.
Instantiation	BAdl implementations are objects that may contain a state. Therefore, it's important whether subsequent similar requests for BAdl instances return the same or different instances of the BAdl implementation. For the instantiation of BAdls, three different modes are supported.	Set to <code>Creating New Instances</code>	This option simplifies the instantiation process.
Multiple Use Indicator	If this indicator is set, several BAdl implementations can be active at the same time.	Set to <code>TRUE</code>	
Use Fallback Indicator	If this indicator is set, the fallback class will be executed in case no other implementation is called.	No restrictions	
Limited Filter Use Indicator	If this checkbox is marked, at most only one filter must be defined in the BAdl definition.	Set to <code>FALSE</code>	It's not allowed to prevent interference with the mandatory filter <code>TENANT_PREFIX</code> .
AMDP BAdl	Option to use a BAdl definition in an ABAP-Managed Database Procedure (AMDP)	Set to <code>FALSE</code>	This option can't be set due to the limitations of our app.
Example Classes	In order to demonstrate how to implement the BAdl method, you may use implementation example classes to provide a template for your customers.	Mandatory	Exactly one example class must be implemented for your customers to refer to. The class should contain meaningful restricted ABAP coding.
BAdl Interface	Defines the methods for this BAdl.	For restrictions, see the next section below	

## Naming Conventions

The description of the BAdI should relate to, or be, a noun. For example, the description of the BAdI that converts A to B should rather be "A To B Conversion" than "Convert A To B". The technical (ABAP) name should be derived from the description (shortened to 30 characters maximum, upper case only). The description and the technical name of the BAdI are visible to the customer.

Do not use technical terms like "Enhancement Spot", "Extensibility", or "Customer" in the description and in the technical name.

For the interface parameter names, use the following naming convention: Use `BUSINESSPARTNER` as importing parameter instead of `IV_BP_ID`. Do not use technical names, abbreviations, or the hungarian notation.

## BAdI Interface

Define the method for the BAdI. Note that the BAdI interface must include the interface `IF_BADI_INTERFACE`, and no other interfaces should be included. Choose a name for the method, and remember to use exactly one method for the interface. Note the following restrictions for the interface **method signature**:

- Use only `IMPORTING` and `CHANGING` parameters. Use `CHANGING` parameters only for the output. This requirement is due to the BAdI framework and limitations of the app.
- Consider `CHANGING` parameters as returning parameters from the key users' point of view, which means that you shouldn't pass values through the parameters.
- Use `TYPE` as the typing method.
- The use of released CDS entities as parameters is allowed.
- The method must have the exception `CX_BLE_RUNTIME_ERROR` in its `RAISING` clause. No other exceptions must be raised. The framework catches all catchable exceptions, logs them and re-throws them as `CX_BLE_RUNTIME_ERROR`.
- The table type is allowed for `IMPORTING` and `CHANGING` parameters. Please use a limited number of fields to reduce the complexity.
- Nested tables are technically allowed for both `IMPORTING` and `CHANGING` parameters, but they should only be used in exceptional cases where their usage simplifies the process.
- Please use only a limited number of `IMPORTING` or `CHANGING` parameters.

### ⓘ Note

Whenever possible, make the method signatures self-contained by using parameter types which you define directly in ABAP within the BAdI interface. Example: `TYPES ty_name TYPE c LENGTH 30`.

## Additional Notes

All ABAP Dictionary types that are used in the parameter typing must be C1-released with the visibility [Use in Key User Apps](#). It's often useful to create wrapper classes to control the set of released data types. Furthermore, any catchable exception raised from the key user coding (directly or indirectly by called methods) will be caught and re-raised in the previous field of `CX_BLE_RUNTIME_ERROR`. This is done by a try-catch

statement that is automatically wrapped around the key user's coding. Variable `IMPLEMENTATION_NAME` of exception `CX_BLE_RUNTIME_ERROR` contains the name of the implementation where the error occurred. If you show messages to the user, remember to always mention this implementation, so that the key user has a chance to find the error and fix it. Adding to that, non-catchable exceptions will still cause a backend dump, as in for example:

`ITAB_DUPLICATE_KEY`: Updating the unique table key `PRIMARY_KEY` resulted in a duplicate entry.

## BAdI Filter Definition

Define a filter for your BAdI definition. Note that filter usage in BAdI implementations is restricted: Filters shouldn't overlap. To make sure that this is not the case, the filters of two distinct implementations of the same BAdI definition should not be true at the same time. See the table below for more restrictions and requirements when defining filters:

Filter Definitions

Option	Description	Recommendation/Restriction
Type	Data type of BAdI filter attribute.	The following types are available: <ul style="list-style-type: none"> <li>I - Integers</li> <li>C - Character type</li> <li>S - String</li> <li>N - Numeric</li> <li>P - Packed (not allowed)</li> </ul>
Description	The short description of the filter.	This field is mandatory. Please keep the short description clear and intuitive for frontend display.
Only constant filter values	The field indicates filters which must be specified as constants when calling them. Therefore, you can already make sure during the compilation of your BAdI whether or not a corresponding BAdI implementation is called. This improves the runtime.	This option can be set depending on business use case. The default is that the indicator is not set.
Value check	It's possible to check the filter values maintained for BAdI implementations.	Set the value check to <code>FALSE</code> . Note that filter value checks are not supported.

### ⓘ Note

When implementing the BAdI, the key user maintains filter values to specify under which conditions the implementation will be called at runtime. The [Custom Logic](#) app only allows the definition of conditions in disjunctive normal form, and only allows comparisons containing a single operator. For example: `(area = 1 AND category > 0 AND volume > 0 AND volume <= 100) OR (area = 1 AND category > 1 AND volume > 100 AND volume <= 199) OR ...`

Key users can maintain the conditions in a form-based editor. At runtime, at most one customer-provided implementation will be executed

## Tenant Prefix Filter

Add a filter called `TENANT_PREFIX` with the type `CHARACTER` to your BAdI definition. This filter ensures that only BAdI implementations for the current tenant are executed at runtime. In your code, obtain the current tenant's prefix by using the ABAP API `cl_ble_api_mt_tenant`. The method `GET_PREFIX` returns the tenant prefix.

```
CLASS zcl_badi_demo DEFINITION PUBLIC FINAL CREATE PUBLIC.
  PUBLIC SECTION.
    METHODS:
      my_method.
  ENDCLASS.

CLASS zcl_badi_demo IMPLEMENTATION.
  METHOD my_method.
    DATA(lo_tenant_api) = cl_ble_api_mt_tenant->create_instance( ).
    DATA(lv_tenant_prefix) = lo_tenant_api->get_prefix( ).

    DATA lo_badi TYPE REF TO z_badi.

    GET BADI lo_badi FILTERS tenant_prefix = lv_tenant_prefix.

    TRY.
      CALL BADI lo_badi->execute.
      CATCH cx_ble_runtime_error INTO DATA(lx_ble_runtime_error).
        " Handle exception.
    ENDTRY.
  ENDMETHOD.
ENDCLASS.
```

### 4.2.10.4.1.3 Releasing the BAdI Definition

#### Context

Maintain the release status for development objects.

#### Procedure

1. Right-click the development object in the *Project Explorer* and choose *API State*. Select *Add Use System-Internally (Contract C1)*.
2. Set the visibility to *Use in Key User Apps*.
3. Maintain the release status in a similar way for the BAdI definition, the BAdI interface, and for all data types that are used in the BAdI method signature.

## 4.2.10.4.1.4 Providing an Example Implementation

Provide an example implementation of your BAdI for your customers. Note that only one example implementation is allowed. The example code must be written in restricted ABAP because this is the template for the key users, and the key users are only allowed to write restricted ABAP code.

As you can only use the ABAP language version [ABAP for Cloud Development](#) in your example class, it could happen that you write code that isn't allowed to be used by key users. To prevent this from happening, please create an implementation of your BAdI in the [Custom Logic](#) app on your test system. If you don't see syntax errors on the UI, your example code is written in restricted ABAP. Have a look at the code reference for key users here: [Statements in ABAP for Key Users \[page 2635\]](#).

## 4.2.10.4.1.5 Registering the BAdI Definition

### Context

To make your BAdI available in the [Custom Logic](#) app, follow the steps below.

### Procedure

1. Right-click on your BAdI definition in the enhancement spot detail screen and choose [Custom Logic](#).
2. Select [Check](#) to verify that your BAdI was implemented correctly. In case error messages appear, fix the root cause of the errors. Then, select [Register](#).
3. Choose a transport request and select [Finish](#). Your BAdI definition has now been registered.
4. To deregister your BAdI, right-click on the BAdI definition and select [Custom Logic](#). Then, choose [Deregister](#). Click [Next](#), choose a transport request and select [Finish](#). Now, the BAdI has been deregistered.

#### ⓘ Note

Do not deregister a BAdI that was already shipped to customers. There could be existing implementations that are no longer visible when the BAdI is gone.

Remember to deregister the BAdI that is registered for [Custom Logic](#) first before deleting it. If you delete the BAdI without deregistering it, you'll receive an error message preventing you from saving the enhancement spot. You can go back to the last active version by closing the enhancement spot without saving.

## 4.2.10.4.2 Configuring Predefined Custom Fields

The predefined extension field framework allows you, as an SAP partner, to equip your SaaS applications with the support of customer-specific extension fields. Customers can extend partner applications with customer-specific extension fields once the application has been deployed on SAP BTP ABAP Environment Multitenancy.

An extensible partner application contains fields which are marked as predefined extension fields. Those fields are part of application but invisible in OData Services by default. You can enable the predefined custom fields in ABAP Developer Tools (ADT) and configure the extension fields using the Fiori app [Configure Predefined Custom Fields](#).

### Related Information

- [Working with Predefined Field Enablings](#)
- [Configure Predefined Custom Fields \[page 2638\]](#)
- [Manage API Snapshots \[page 2556\]](#)

## 4.2.11 Released Components and Objects

To establish lifecycle independence of SAP code and your own applications as well as extensions built in the ABAP environment, you can only use released components and reuse services, together with a cloud-optimized scope of the ABAP language. Those components and reuse services form the public, upgrade-stable interface of the ABAP environment. Any other object or service that is not released can't be used for your own development.

In this chapter, you can learn more about released APIs from the ABAP language, technical APIs, and released services.

### 4.2.11.1 ABAP Language

Get an overview of all the objects and services of the ABAP programming language that were released for use in the ABAP environment.

You can use them for basic programming techniques such as data handling and database access. Note that not all objects and services that you know from the ABAP on-premise environment are available (or available in the same way).

## More Information and Help

For more detailed information about the objects and services listed here, go to the [ABAP Keyword Documentation](#) (ABAP language help) or, in the relevant development object in ABAP Development Tools (ADT), directly to ABAP Doc.

To access the ABAP language help from the source code editor, position your cursor on an ABAP statement for which you need help, and press **F1**.

### 4.2.11.1.1 ADT Class Execution

Easily execute ABAP classes directly in ABAP Development Tools (ADT) without having to launch a service.

#### Sample Code

```
CLASS zcl_example_class DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_example_class IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    out->write( 'Hello World!' ).
  ENDMETHOD.
ENDCLASS.
```

For more information, see the ABAP Doc comments of the `IF_OO_ADT_CLASSRUN` interface in ABAP Development Tools (ADT).

### 4.2.11.1.2 ABAP Managed Database Procedures (AMDP)

Use this class-based framework for managing and calling stored procedures or database procedures as AMDP procedures.

For more information, see the [ABAP Keyword Documentation](#).

### 4.2.11.1.3 Arithmetic

Get an overview of the service classes that support arithmetic features such as conversions and random number generators.

For more information about numeric calculations, see the [ABAP Keyword Documentation](#).

#### **4.2.11.1.4 Compression/Decompression**

You can compress and decompress data using the `gzip` format.

For more information, see the ABAP Doc comments of the classes in package `SABP_GZIP` in ABAP Development Tools (ADT).

#### **4.2.11.1.5 Database Access**

You can use locators and streams to access large binary or large character objects (BLOB and CLOB) in the database.

For more information, see the [ABAP Keyword Documentation](#).

#### **4.2.11.1.6 Date/Time and Time Stamp Processing**

To handle and convert time stamps, use the following classes:

- `CL_ABAP_DATFM`
- `CL_ABAP_TIMEFM`
- `CL_ABAP_UTCLONG`
- `CL_ABAP_TSTMP`

For more information, see the ABAP Doc comments for these classes.

For more information about handling time stamps, see [System Class for Time Stamps in Packed Numbers](#) and [System Class for Time Stamp Fields](#) in the ABAP Keyword Documentation.

#### **4.2.11.1.7 Parallel Processing**

To start parallel tasks, use class `CL_ABAP_PARALLEL`.

### **Context**

In an ABAP system, dialog processes are intended for processing dialog requests. In most cases, a large number of dialog processes is unused and you want to assign these free processes to parallel processing.

## Procedure

Specify how many and which of the free dialog processes are used for parallel tasks when you create an instance of `CL_ABAP_PARALLEL`:

### ⓘ Note

Keep in mind that two of the free dialog processes on a server are always retained by the system for dialog requests.

- If you set the parameter `P_LOCAL_SERVER` to the value '`X`', only processes of the current server are used. By default, the dialog processes of all servers of the current SAP system are used.
- With parameter `P_PERCENTAGE`, you can set the maximum percentage of free dialog processes to be used. If nothing is specified here, a value of 50 percent is used.

Transporting the data into a free dialog process may take some time. Therefore, it's not efficient to parallelize very short-running operations. The time limit is around 200 milliseconds.

### → Recommendation

Shorter operations should be bundled, that is, several short operations should be executed in one task.

## Error Handling

When the framework is called, an internal table with the information on the tasks to be executed is transferred for each variant.

The result is a table that has the same number of entries as the input table, regardless of whether the tasks were actually executed. The output table also has the same order. A structure is returned as the result for each task. If the task was successful, the results of the processing are returned. In the case of the `RUNT_INST` method, for example, a result instance is returned in the `INST` component. If the processing was not successful for any reason, a message is returned instead. The corresponding field is the `MESSAGE` component in the result structure.

Messages from the framework are returned in this message field. For example, if a task is canceled due to a timeout, the following message is returned:

```
Time limit exceeded
```

If a task was aborted with a runtime error, the short text of the corresponding runtime error is provided as a message.

## Related Information

For more information, see the ABAP Doc comments of the `CL_ABAP_PARALLEL` class in ABAP Development Tools (ADT).

## 4.2.11.1.8 Runtime Type Information

Implement the Runtime Type Services (RTTS) using a hierarchy of type description classes that contain the methods for Runtime Type Creation (RTTC) and Runtime Type Identification (RTTI).

Using these system classes, you can do the following:

- Determine type information of existing instances and type names in the ABAP type system at runtime.
- Define new data types at runtime.

For more information, see the ABAP Doc comments of class `CL_ABAP_TYPEDESCR` and its subclasses in the ABAP Development Tools (ADT).

## 4.2.11.1.9 Security

To support secure dynamic programming in ABAP, use class `CL_ABAP_DYN_PRG`.

For more information, see the ABAP Doc comments of this class in ABAP Development Tools (ADT).

## 4.2.11.1.10 String Processing

Use the following classes to process strings:

- `CL_ABAP_[ STRING | CHAR ]_UTILITIES`  
Provides auxiliary functions for string/character processing
- `CL_ABAP_CONV_CODEPAGE`  
For code page conversion of strings/characters
- `CL_ABAP_[ REGEX | MATCHER ]`  
For regular expression processing  
For more information about regular expressions, see the [ABAP Keyword Documentation](#).

For more information about these classes, see their ABAP Doc comments in ABAP Development Tools (ADT).

## 4.2.11.1.11 Structure Processing

You can use the system class `CL_ABAP_CORRESPONDING` to assign components between structures or between internal tables with dynamically specified mapping rules. Methods are available for simple assignments, assignments of default values, and for using a lookup table.

For more information, see the [ABAP Keyword Documentation](#).

## 4.2.11.1.12 System Information

To get technical information such as the current user, language, or time zone, use class `CL_ABAP_CONTEXT_INFO`.

To query for released objects, use the CDS view `I_APISForSAPCloudPlatform`.

For more information, see the relevant ABAP Doc comments in ABAP Development Tools (ADT).

## 4.2.11.1.13 XML/XSLT/ST

You can handle and process XML data using XSLT or ST programs.

For more information about how XML data can be created and read in ABAP, see the [ABAP Keyword Documentation](#).

## 4.2.11.1.14 Access to System Structure SY

Access to the system structure `SY` is restricted to read access to the following components:

`BATCH`, `DBCNT`, `FDPOS`, `INDEX`, `LANGU`, `MSGID`, `MSGNO`, `MSGTY`, `MSGV1`, `MSGV2`, `MSGV3`, `MSGV4`, `SUBRC`, `TABIX`, and `UNAME`.

### ⓘ Note

Access to all other components is **not** allowed because they are related to either obsolete or unsupported features.

For more information, see [ABAP System Fields](#) in the ABAP Keyword Documentation.

You can use class `CL_ABAP_CONTEXT_INFO` to retrieve information about the user session, for example, technical user name, business user name, time zone, and so on. The built-in function `utclong_current` generates a UTC time stamp from the current system time and the current system date in accordance with POSIX standards.

## 4.2.11.2 Change Document Solution

You can document changes made to a commercial object, such as the time, the content and the way changes are made, by logging these changes in a change document.

### ⚡ Example

You can use the change document to simplify the change history analysis for auditing in [Financial Accounting](#).

Every application object type has its own change document object type, which is called the *Change Document Object*. To log changes to a commercial object in a change document, you must define the *Change Document Object* for the commercial object type. The *Change Document Object* definition contains the tables which represent a commercial object in the system.

### ① Note

- Specify for each table, whether a commercial object contains only one (single case) or several (multiple case) records. For example, an order contains an order header and several order items. In general, one record for the order header and several records for the order items are passed to the change document creation when an order is changed.
- If a table contains fields with values referring to units and currency fields, the associated table, containing these units and currencies, can also be specified.
- The object ID identifies a given commercial object. You can retrieve all changes made to a commercial object using this key.

## Change Document Structure

- Change Document Header

The change document header is the part of the change document which contains general information about the changes made. The document header is identified by object class name, the object ID and a change document number (automatically provided by the runtime system). The change document header contains information about who made the changes, when they were made, what type of changes were made (update, insert or delete operation) and if they were real or planned changes. Starting with Release 6.20 information about the system language and the length of the table key of the documented table are saved here as well. The header data is stored in table CDHDR.

- Change Document Item

The change document item contains the old and new values of a field for a particular change, and a change flag, that presents the kind of change. The change flag can take the following values:

- U = Update

Changed data. (Log entry for each changed field which was flagged in the Dictionary as *change document relevant*.)

- I = Insert

Data Inserted.

Changes: Log entry for the whole table record.

Planned changes: Log entry for each table record field.

- D = Delete

Data were deleted (log entry for the whole table record).

- E = Individual field documentation at Delete

Delete a table record with field documentation.

One log entry per field of the deleted table entry.

- J = Individual field documentation at Insert

Insert a table record with field documentation.

One log entry per field of the inserted table entry.

## 4.2.11.2.1 Authorization Checks

### Authorization Check for Change Document Object Maintenance

There is an authority check during all methods of Change Document Object Maintenance. Use method `IF_CHDO_OBJECT_TOOLS_REL~CHECK_AUTHORIZATION` to run the authorization check separately.

Use method `IF_CHDO_OBJECT_TOOLS_REL~CHECK_AUTHORIZATION` to run an additional authorization check. `IV_OBJECT`, `IV_DEVCLASS` and `IV_ACTIVITY` get passed as import parameters. The return parameter `RV_IS_AUTHORIZED` must be set to `ABAP_TRUE` if the check is successful.

Import Parameters

Parameter Name	Field Name	Value Help
IV_OBJECT		Name of Change Document Object
IV_ACTIVITY		Activity to be checked. Existing values: <ul style="list-style-type: none"><li>• '01' – Create</li><li>• '02' – Change</li><li>• '03' – Read</li><li>• '06' = delete</li></ul>
IV_DEVCLASS		

Return Parameters

Parameter Name	Field Name	Value Help
RV_IS_AUTHORIZED		ABAP_TRUE if the authority check was successfully

#### ↔ Sample Code

```
CLASS zcl_chdo_test_auth DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_test_auth IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: lv_is_authorized type abap_bool.
    TRY.
      " iv_object : Change document object name
      " it_activity : Activity to be checked. Possible values '01' = create,
      "                                     '02' = change,
      "                                     '03' = read,
      "                                     '06' = delete
      " it_devclass : development class of change document object
      cl_chdo_object_tools_rel=>if_chdo_object_tools_rel~check_authorization(
```

```

EXPORTING
    iv_object      = 'ZCHDO_TEST'
    iv_activity    = '03'
    iv_devclass    = 'ZLOCAL'
RECEIVING
    rv_is_authorized = lv_is_authorized
).
ENDTRY.
IF lv_is_authorized IS INITIAL.
    out->write( |Exception occurred: authorization error.| ).
ELSE.
    out->write( |Activity can be performed on the change document
object.| ).
ENDIF.
ENDMETHOD.
ENDCLASS.

```

## Authorization Check for Reading Change Documents

When a change document object is generated, you receive the <name\_space>CL\_<change document object name>\_CHDO class with method IF\_CHDO\_ENHANCEMENTS~AUTHORITY\_CHECK without implementation. You can create your own authority check for reading change documents written for this change document object. The authority check for reading change documents are successful if parameter RV\_IS\_AUTHORIZED = 'X' is returned.

### 4.2.11.2.2 Managing Change Document Objects

To maintain a change document object during design time, the ABAP Development Tools UI editor for change document objects can be used. For more information, see [Working with Change Document Objects](#).

Use class CL\_CHDO\_OBJECT\_TOOLS\_REL to maintain change document objects.

For more information, see

- [Creating a Change Document Object \[page 1435\]](#)
- [Updating a Change Document Object \[page 1440\]](#)
- [Deleting a Change Document Object \[page 1444\]](#)
- [Reading a Change Document Object Definition \[page 1446\]](#)

### Related Information

[Working with Change Document Objects](#)

## 4.2.11.2.2.1 Creating a Change Document Object

When creating a change document object, the following naming-rules apply:

- For customer objects, start the name with a 'Z' or a 'Y'. For more information, see SAP Note 16466.
- If a name space is requested, use an object starting with namespace as object name.
- Keep in mind that the change document object name including the name space has a maximum length of 15 characters. That means that if the name space has 10 characters, only 5 characters are left for the change document object name.

### Process

Use method `IF_CHDO_OBJECT_TOOLS_REL~CREATE_AND_GENERATE_OBJECT` to create and generate change document objects.

The name of the object is assigned using the import parameter `IV_OBJECT`. Object details and generation information are passed using the internal tables `IT_CD_OBJECT_DEF` (the object definition), `IT_CDOBJECT_TEXT` (object texts) and `IS_CD_OBJECT_GEN` (generation information).

Once generated, the class (`<name space>CL_<change document object name>_CHDO`) with methods `WRITE` and `IF_CHDO_ENHANCEMENTS~AUTHORITY_CHECK` are created. `IV_CL_OVERWRITE` can be used to specify whether an existing class with the specified name can be overwritten or not. If a class already exists and can't be overwritten, no changes will be made. Changes are saved in the transport request (`IV_CORRNR`).

Method `AUTHORITY_CHECK` of interface '`IF_CHDO_ENHANCEMENTS`' must be implemented later on in order to enable an authorization check for reading change documents.

The export parameter `ET_ERRORS` is used to return all generation messages (messages from message class CD). Any syntax errors in the generated class are provided using `ET_SYNT_ERROR` (with long text if applicable (`ET_SYNT_ERROR_LONG`)).

#### Import Parameters

Parameter Name	Field Name	Value Help
<code>IV_OBJECT</code>		Name of change document object
<code>IT_CD_OBJECT_DEF</code>		Change document object definition
	<code>TABNAME</code>	Name of the table as defined in the dictionary
	<code>MULTCASE</code>	If more than one record for a particular table is to be documented during a single CREATE/UPDATE/DELETE operation, the value <code>MULTCASE</code> should be ' <code>ABAP_TRUE</code> ' or ' <code>X</code> ' (multiple case). If no value is provided a single record can be documented and passed in a work area (single case).

Parameter Name	Field Name	Value Help
	DOCUDEL	<p>DOCUDEL = SPACE</p> <p>The deletion of a table entry will be documented in one change document item using the table key to identify the table entry deleted. The field FNAME will be filled with 'KEY'.</p> <p>DOCUDEL = 'X'</p> <p>Each change document relevant field value of the table entry will be documented individually in its own change document item.</p> <p>The change indicator is 'E' instead of 'D'.</p>
	DOCUINS	<p>DOCUINS = SPACE</p> <p>The insert of a table entry will be documented in one change document item using the table key to identify the inserted table entry. The field FNAME will be filled with 'KEY'.</p> <p>DOCUINS = 'X'</p> <p>Each change document relevant field value of the table entry will be documented individually in its own change document item.</p> <p>The change indicator is 'J' instead of 'I'.</p>
	REFNAME	<p>If fields of table TABNAME reference unit or currency field values from another table, the name of that table must be passed here to document its values as well. Only one reference table can be used for table TABNAME.</p> <p>In single case, the referenced entry from table REFNAME is passed as two additional work areas (old, new). In multiple case, the import tables (old, new) are enhanced to include the referenced structure of table REFNAME.</p>

Parameter Name	Field Name	Value Help
	DOCUD_IF	If you want to document the value of a field even though it is initial when data is deleted, mark this field. If it is not marked, log entries will only be written if the field values are not initial when they are deleted.  Be aware that a lot of additional change documents may be written, if you choose this option. Only mark this flag if it is required.
	DOCUI_IF	If you want to document the value of a field even though it is initial when data is inserted, mark this field. If it is not marked, log entries will only be written if the field values are not initial when they are inserted.  Be aware that a lot of additional change documents may be written, if you choose this option. Only mark this flag, if it is required.
IT_CD_OBJECT_TEXT		Object texts for change document object
	LANG_KEY	Language key of the text
	OBJECT_TEXT	Descriptive short text for the change document object
IS_CD_OBJECT_GEN		Change document object generation information
	AUTHOR	User who performs the generation
	UPDNAME	User who performs the change
	CHANGE_DATE	Date of change
	CHANGE_TIME	Time of change
	TEXTCASE	Special Text Handling flag  Select this field to log long text changes. The old and new status of long texts is not logged. Only the fact that they have been changed is noted.

Parameter Name	Field Name	Value Help
	DEVCLASS	Change document object package
IV_CL_OVERWRITE		Whether generated class should overwrite an already existing class. Value 'X' means an existing class will be overwritten.
IV_CORRNR		Transport request where changes should be logged
Export Parameters		
Parameter Name	Field Name	Value Help
ET_ERRORS		
	kind	Message type (empty means information message, ,E-, means error)
	msgid	Message class (CD)
	msgnr	Message ID
	v1	Variable to message
	v2	Variable to message
	v3	Variable to message
	v4	Variable to message
	text	Short text of the message
ET_SYNT_ERRORS		Syntax errors raised during generation of class. Check syntax of generated class directly.
ET_SYNT_ERROR_LONG		Syntax errors raised during generation of class. Check syntax of generated class directly.

#### ↔ Sample Code

```

CLASS zcl_chdo_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_test IMPLEMENTATION.

```

```

METHOD if_oo_adt_classrun~main.
DATA:
ls_error      TYPE abap_bool,
rt_errors     TYPE if_chdo_object_tools_rel=>ty_tr_error_tab,
lt_errors_err TYPE LINE OF if_chdo_object_tools_rel=>ty_tr_error_tab,
p_it_tcdob   TYPE if_chdo_object_tools_rel=>ty_tcdobdef_tab,
p_it_tcdobt  TYPE if_chdo_object_tools_rel=>ty_tcdobtext_tab,
p_it_tcdrp   TYPE if_chdo_object_tools_rel=>ty_tcdgen,
lt_tcdobt    TYPE if_chdo_object_tools_rel=>ty_tcdobt_tabtyp,
ls_tcdob     TYPE LINE OF if_chdo_object_tools_rel=>ty_tcdobdef_tab,
ls_tcdobt   TYPE LINE OF if_chdo_object_tools_rel=>ty_tcdobtext_tab.
data: lr_err          TYPE REF TO cx_chdo_generation_error.
ls_tcdob-tabname = 'ZCHDO'.
ls_tcdob-multcase = ''.
ls_tcdob-docudel = ''.
ls_tcdob-docuins = ''.
ls_tcdob-docud_if = ''.
APPEND ls_tcdob TO p_it_tcdob.
ls_tcdobt-lang_key = 'D'.
ls_tcdobt-object_text = 'Single Case'.
APPEND ls_tcdobt TO p_it_tcdobt.
p_it_tcdrp-author = 'X11'.
p_it_tcdrp-textcase = 'X'.
p_it_tcdrp-devclass = '<development class>'.
CLEAR: rt_errors, lr_err.
TRY.

cl_chdo_object_tools_rel=>if_chdo_object_tools_rel~create_and_generate_object(
  EXPORTING
    iv_object      = 'ZCHDO_TEST' " change document object name
    it_cd_object_def = p_it_tcdob " change document object
definition
    it_cd_object_text = p_it_tcdobt " change document object text
    is_cd_object_gen = p_it_tcdrp " change document object
generation info
    iv_cl_overwrite = 'X'           " class overwrite flag
    iv_corrnr       = '<transport_request>' " transport request
number
  IMPORTING
    et_errors      = rt_errors      " generation message table
*     et_synt_errors =
*     et_synt_error_long =
).
CATCH cx_chdo_generation_error into lr_err.
  out->write( |Exception occurred: { lr_err->get_text( ) }| ).
  ls_error = 'X'.
ENDTRY.
IF ls_error IS INITIAL.
  READ TABLE rt_errors WITH KEY kind = 'E-'
    INTO lt_errors_err.
  IF sy-subrc IS INITIAL.
    out->write( |Exception occurred: { lt_errors_err-text } | ).
  ELSE.
    out->write( |Change document object created and generated | ).
  ENDIF.
ENDIF.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.2.2.2 Updating a Change Document Object

Use method `IF_CHDO_OBJECT_TOOLS_REL~UPDATE_OBJECT` to modify and regenerate change document objects.

The name of the object is assigned using the import parameter `IV_OBJECT`. The object details and generation information are passed using the internal tables `IT_CD_OBJECT_DEF` (the object definition), `IT_CD_OBJECT_TEXT` (the object texts) and `IS_CD_OBJECT_GEN` (the generation information).

If the internal tables are not filled, the object definition is read directly from the database tables `TCDOB` and `TADIR` and the generation information is read directly from database table `TCDRP`. If only parameter `IS_CD_OBJECT_GEN` is passed, the change document object will be newly generated without changing the change document object definition. Use this method when the structure of a table, that belongs to the change document object, was changed.

Once generated, a class (`<name space>CL_<change document object name>_CHDO`) with methods `WRITE` and `IF_CHDO_ENHANCEMENTS~AUTHORITY_CHECK` are created. You can use `IV_CL_OVERWRITE` to specify whether an existing class can be overwritten with the specified name or not. If a class already exists and can't be overwritten, no changes will be made. Changes made to the class are saved in the transport request (`IV_CORRNR`).

Method `AUTHORITY_CHECK` of interface '`IF_CHDO_ENHANCEMENTS`' must be implemented later on in order to enable an authorization check for reading change documents.

The export parameter `ET_ERRORS` is used to return all generation messages (messages from message class `CD`). Any syntax errors in the generated class are provided using `ET_SYNT_ERROR` (with long text if applicable (`ET_SYNT_ERROR_LONG`)).

### Import Parameters

Parameter Name	Field Name	Value Help
<code>IV_OBJECT</code>		Name of change document object
<code>IT_CD_OBJECT_DEF</code>		Change document object definition
	<code>TABNAME</code>	Name of the table as defined in the dictionary
	<code>MULTCASE</code>	If more than one record for a particular table is to be documented during a single <code>CREATE/UPDATE/DELETE</code> operation, the value <code>MULTICASE</code> should be ' <code>ABAP_TRUE</code> ' or ' <code>X</code> ' (multiple case). If no value is provided a single record can be documented and passed in a work area (single case).

Parameter Name	Field Name	Value Help
	DOCUDEL	<p>DOCUDEL = SPACE</p> <p>The deletion of a table entry will be documented in one change document item using the table key to identify the table entry deleted. The field FNAME will be filled with 'KEY'.</p> <p>DOCUDEL = 'X'</p> <p>Each change document relevant field value of the table entry will be documented individually in its own change document item.</p> <p>The change indicator is 'E' instead of 'D'.</p>
	DOCUINS	<p>DOCUINS = SPACE</p> <p>The insert of a table entry will be documented in one change document item using the table key to identify the inserted table entry. The field FNAME will be filled with 'KEY'.</p> <p>DOCUINS = 'X'</p> <p>Each change document relevant field value of the table entry will be documented individually in its own change document item.</p> <p>The change indicator is 'J' instead of 'I'.</p>
	REFNAME	<p>If fields of table TABNAME reference unit or currency field values from another table, the name of that table has to be passed here to document its values as well.</p> <p>Only one reference table can be used for table TABNAME.</p> <p>In single case, the referenced entry from table REFNAME is passed as two additional work areas (old, new). In multiple case, the import tables (old, new) are enhanced to include the referenced structure of table REFNAME.</p>

Parameter Name	Field Name	Value Help
	DOCUD_IF	If you want to document the value of a field even though it is initial when data is deleted, mark this field. If it is not marked, log entries will only be written if the field values are not initial when they are deleted.  Be aware that a lot of additional change documents may be written, if you choose this option. Only mark this flag if it is required.
	DOCUI_IF	If you want to document the value of a field even though it is initial when data is inserted, mark this field. If it is not marked, log entries will only be written if the field values are not initial when they are inserted.  Be aware that a lot of additional change documents may be written, if you choose this option. Only mark this flag, if it is required.
IT_CD_OBJECT_TEXT		Object texts for change document object
	LANG_KEY	Language key of the text
	OBJECT_TEXT	Descriptive short text for the change document object
IS_CD_OBJECT_GEN		Change document object generation information
	AUTHOR	User who performs the generation
	UPDNAME	User who performs the change
	CHANGE_DATE	Date of change
	CHANGE_TIME	Time of change
	TEXTCASE	Special Text Handling flag  Select this field to log long text changes. The old and new status of long texts is not logged. Only the fact that they have been changed is noted.

Parameter Name	Field Name	Value Help
	DEVCLASS	Change document object package
IV_CL_OVERWRITE		Whether generated class should overwrite an already existing class. Value 'X' means an existing class will be overwritten.
IV_CORRNR		Transport request where changes should be logged
Export Parameters		
Parameter Name	Field Name	Value Help
ET_ERRORS		
	kind	Message type (empty means information message, ,E-, means error)
	msgid	Message class (CD)
	msgnr	Message ID
	v1	Variable to message
	v2	Variable to message
	v3	Variable to message
	v4	Variable to message
	text	Short text of the message
ET_SYNT_ERRORS		Syntax errors raised during generation of class. Check syntax of generated class directly.
ET_SYNT_ERROR_LONG		Syntax errors raised during generation of class. Check syntax of generated class directly.

### ↳ Sample Code

```

CLASS zcl_chdo_update_object DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_update_object IMPLEMENTATION.

```

```

METHOD if_oo_adt_classrun~main.
DATA:
    rt_errors          TYPE if_chdo_object_tools_rel=>TY_TR_ERROR_TAB,
    lt_errors_err      TYPE LINE OF
if_chdo_object_tools_rel=>TY_TR_ERROR_TAB,
    lt_tcDOB          TYPE iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBDEF_TAB,
    P_IT_TCDOB         TYPE iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBDEF_TAB,
    P_IT_TCDOBt        TYPE iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBTEXT_TAB,
    P_IT_TCDRP         TYPE iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDGEN,
    lt_tcDOBt          TYPE iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBTEXT_TAB,
    ls_tcDOB           TYPE LINE OF
iF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBDEF_TAB,
    ls_tcDOBt          TYPE LINE OF
IF_CHDO_OBJECT_TOOLS_REL=>TY_TCDOBTEXT_TAB.
    data: lr_err          TYPE REF TO cx_chdo_generation_error.
    ls_tcDOB-tabname = 'ZCHDO'.
    ls_tcDOB-multcase = 'X'.
    ls_tcDOB-docudel = ''.
    ls_tcDOB-docuins = ''.
    ls_tcDOB-docud_if = ''.
APPEND ls_tcDOB TO p_it_tcDOB.
    ls_tcDOB-lang_key = 'D'.
    ls_tcDOB-object_text = 'Single Case: Update'.
APPEND ls_tcDOB TO p_it_tcDOBt.
    p_it_tcDRP-author = sy-uname.
    p_it_tcDRP-textcase = 'X'.
    p_it_tcDRP-devclass = <development class>.
CLEAR: rt_errors, lr_err.
TRY.
    cl_chdo_object_tools_rel=>if_chdo_object_tools_rel~update_object(
        EXPORTING
            iv_object      = 'ZCHDO_TEST'
            it_cd_object_def = p_it_tcDOB
            it_cd_object_text = p_it_tcDOBt
            is_cd_object_gen = p_it_tcDRP
            iv_cl_overwrite = 'X'
            iv_corrnr       = '<transport request>'
        IMPORTING
            et_errors      = rt_errors
    ).
    CATCH cx_chdo_generation_error INTO lr_err.
ENDTRY.
    IF lr_err IS INITIAL.
        READ TABLE rt_errors WITH KEY kind = 'E-'
            INTO lt_errors_err.
        IF sy-subrc IS INITIAL.
            out->write( |Exception occurred: { lt_errors_err-text } | ).
        ELSE.
            out->write( |Change document object updated | ).
        ENDIF.
    ENDIF.
ENDMETHOD.
ENDCLASS.

```

### 4.2.11.2.2.3 Deleting a Change Document Object

Use method `IF_CHDO_OBJECT_TOOLS_REL~DELETE_OBJECT` to delete change document objects. The name of the object is assigned using the import parameter `IV_OBJECT`.

Furthermore, the import parameter `IV_DEL_CL_WHEN_USED` determines, if the class of the change document object should be deleted (if value passed is `ABAP_TRUE`) or not (value `ABAP_FALSE`) when it is still being used. The changes made to the class are saved in the transport request (`IV_CORRNR`).

The export parameter `ET_ERRORS` is used to return all deletion messages (messages from message class CD).

The object will be deleted if no exception and no error messages of kind 'E-' were raised.

#### Import Parameters

Parameter Name	Field Name	Value Help
<code>IV_OBJECT</code>		Change document object name
<code>IV_CORRNR</code>		Transport request where deletion should be logged
<code>IV_DEL_CL_WHEN_USED</code>		Delete generated class "CL_<change document object name>_CHDO" even if it is currently used.

#### Export Parameters

Parameter Name	Field Name	Value Help
<code>ET_ERRORS</code>		
	<code>kind</code>	Message type (empty means information message, ,E-, means error)
	<code>msgid</code>	Message class (CD)
	<code>msgnr</code>	Message ID
	<code>v1</code>	Variable to message
	<code>v2</code>	Variable to message
	<code>v3</code>	Variable to message
	<code>v4</code>	Variable to message
	<code>text</code>	Short text of the message

#### ↔ Sample Code

```

CLASS zcl_chdo_delete_object DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.
CLASS zcl_chdo_delete_object IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    data: ls_error          TYPE abap_bool,

```

```

        lr_err           TYPE REF TO cx_chdo_generation_error,
        lt_errors_err    TYPE LINE OF
if_chdo_object_tools_rel=>ty_tr_error_tab,
        rt_errors        TYPE
if_chdo_object_tools_rel=>ty_tr_error_tab.
TRY.
cl_chdo_object_tools_rel=>if_chdo_object_tools_rel~delete_object(
EXPORTING
        iv_object          = 'ZCHDO_TEST'      " change document object name
        iv_corrnr         = '<transport_request>'   " transport request
number
        iv_del_cl_when_used = 'X'           " delete class even when it
is used
IMPORTING
        et_errors          = rt_errors       " messages from deletion
).
CATCH cx_chdo_generation_error into lr_err.
        out->write( |Exception occurred: { lr_err->get_text( ) }| | ).
        ls_error = 'X'.
ENDTRY.
IF ls_error IS INITIAL.
    READ TABLE rt_errors WITH KEY kind = 'E-'
        INTO lt_errors_err.
    IF sy-subrc IS INITIAL.
        out->write( |Exception occurred: { lt_errors_err-text } | | ).
    ELSE.
        out->write( |Change document object deleted | |).
    ENDIF.
ENDIF.
ENDMETHOD.
ENDCLASS.

```

#### 4.2.11.2.2.4 Reading a Change Document Object Definition

Use method `IF_CHDO_OBJECT_TOOLS_REL~READ_OBJECT` to read a change document object definition.

The name of the object is assigned using the import parameter `IV_OBJECT`. The information is returned using the export parameter `ET_OBJECT_INFO`.

Import Parameter

Parameter Name	Field Name	Value Help
IV_OBJECT		Change document object name

Export Parameters

Parameter Name	Field Name	Value Help
ET_OBJECT_INFO	ET_OBJECT_INFO	Name of the change document object
	ET_OBJECT_INFO	Generation information counter for field gen_type

Parameter Name	Field Name	Value Help
	ET_OBJECT_INFO	Generation information type technical name (DEFINITION, GENERATION, CLASS)
	ET_OBJECT_INFO	Generation information type text (Definition, Generate, Class details)
	ET_OBJECT_INFO	Line counter for field name
	ET_OBJECT_INFO	Information long text
	ET_OBJECT_INFO	Information technical name
	ET_OBJECT_INFO	Value of the change document object
	OBJECTCLASS	Name of the change document object
	GNR	Generation information counter for field gen_type
	GEN_TYPE	Generation information type technical name (DEFINITION, GENERATION, CLASS)
	GROUP	Generation information type text (Definition, Generate, Class details)
	ZNR	Line counter
	INFOTEXT	Information long text
	OBJ_TYPE	Information technical name
	NAME	Information value (e.g. table name, package, user name, class name)

### ↔ Sample Code

```

CLASS zcl_chdo_read_object DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_read_object IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: pt_object_info TYPE if_chdo_object_tools_rel=>tty_object_info.
    data: lr_err           TYPE REF TO cx_chdo_generation_error.
    TRY.
      cl_chdo_object_tools_rel=>if_chdo_object_tools_rel~read_object(
        EXPORTING

```

```

        iv_object      = 'ZCHDO_TEST'      " change document object name
IMPORTING
    et_object_info = pt_object_info    " change document object details
).
CATCH cx_chdo_generation_error INTO lr_err.
    out->write( |Exception occurred: { lr_err->get_text( ) }| ).
ENDTRY.
ENDMETHOD.
ENDCLASS.
```

### 4.2.11.2.3 Writing Change Documents

The `WRITE` method in the generated class `CL_<change document object name>_CHDO` creates change documents from the object-specific update for an object ID.

If the [Fiori Reuse Library for Change Document Solution](#) will be used to display the change documents, time of change (UTIME) and date of change (UDATE) need to be provided in system time zone.

Import Parameters

Parameter Name	Value Help
OBJECTID	Identifier of the changed application object
UTIME	Time of change (If the <a href="#">Fiori Reuse Library for Change Document Solution</a> is used provide information in system time zone.)
UDATE	Date of change (If the <a href="#">Fiori Reuse Library for Change Document Solution</a> is used provide information in system time zone.)
USERNAME	User name of the person responsible in change document
PLANNED_CHANGE_NUMBER	Planned change number (only for writing planned changes)
OBJECT_CHANGE_INDICATOR	Type of Change to Application Object. Possible values are: <ul style="list-style-type: none"> <li>• U - Change</li> <li>• I - Insert</li> <li>• D - Delete</li> </ul>
PLANNED_OR_REAL_CHANGES	You can use this parameter to control whether the changes to be logged are actual or planned changes.  Available values: <ul style="list-style-type: none"> <li>• "R" actual (real) changes -"P" planned changes</li> <li>• if there is no plan number: actual change if there is a plan number: planned change</li> </ul>

Parameter Name	Value Help
NO_CHANGE_POINTERS	'X': no change pointers will be written. Change pointers are used for Application Link Enabling (ALE) master data scenarios. Currently, this parameter is obsolete for SAP Business Technology Platform (SAP BTP), ABAP Environment.
ICDTXT_<change document object name>	In this structure, the change document-relevant texts are collected with the corresponding specifications: <ul style="list-style-type: none"> <li>• TEILOBJID: Key of changed table row</li> <li>• TEXTART: Text type of changed text</li> <li>• TEXTSPR: Language Key</li> <li>• UPDKZ: Change flag for table row: D(elete), I(nsert) or U(pdate)</li> </ul>
UPD_ICDTXT_<change document object name>	Change flag for text table: <ul style="list-style-type: none"> <li>• " " (Space): Table is not considered.</li> <li>• "U": Table is considered</li> </ul>
O_<table name>	Object-specific parameter for single case tables with the same table structure as <table name>. The workarea must contain the original data.
N_<table name>	Object-specific parameter for single case tables with the same table structure as <table name>. The workarea must contain the original data.
Y<table name>	Object-specific parameter for multiple case tables.  The table must contain the original version of the changed or deleted records. The structure consists of the table, as specified in the change document object definition under table name, a processing flag (TYPE C, length 1)

Parameter Name	Value Help
X<table name>	<p>The table must contain the current version of the changed or created records. The structure is the same as Y&lt;table name&gt; (see above). It is not necessary to set the processing flag to values &lt;&gt; space for standard behavior. The following values for the processing flag exist:</p> <ul style="list-style-type: none"><li>• "I" (INSERT)Records were created, or table records were deleted, then a record with the same key was created in the same transaction, and this is to be documented as "Delete" and "Create" (special case), not as "Change".</li><li>• "U" or " " (space) (UPDATE)</li></ul> <p>The parameter UPD_&lt;table name&gt; (see below) initially determines whether the record is new or changed. The processing flag is only checked when - with the parameter value "U" - the following key comparison between the two tables x&lt;table name&gt; and y&lt;table name&gt; finds two records with the same key.</p>

Parameter Name	Value Help
UPD_<table name >	<p>With this flag, you determine the processing logic. The following values exist:</p> <ul style="list-style-type: none"> <li>• "D" (DELETE) A change document item is to be created for each record in Y&lt;table name &gt;. X&lt;table name&gt; is not processed.</li> <li>• "I" (INSERT) A change document item is to be created for each record in X&lt;table name&gt;. Y&lt;table name&gt; is not processed.</li> <li>• "U" (UPDATE) The keys of Y&lt;table name&gt; and X&lt;table name&gt; are compared. The following cases are distinguished: <ul style="list-style-type: none"> <li>• Record exists in Y&lt;table name&gt; but not in X&lt;table name&gt;: Change document items are to be created for the record in Y&lt;table name&gt; which is to be deleted.</li> <li>• Record exists in X&lt;table name&gt; but not in Y&lt;table name&gt;: A change document item is to be created for the records in X&lt;table name&gt; which are to be flagged as created.</li> <li>• Record exists in both Y&lt;table name&gt; and X&lt;table name&gt; and processing flag in X&lt;table name&gt; record is "U" or space: A change document item is created for each changed field which is defined as change document-relevant in the dictionary.</li> <li>• Record exists in both Y&lt;table name&gt; and X&lt;table name&gt; and processing flag in X&lt;table name&gt; record is "I": Change document item is to be created for the record in Y&lt;table name&gt; which is to be deleted and change document item is to be created for the record in X&lt;table name&gt; which is to be flagged as created.</li> <li>• " " (space, no processing) X&lt;table name&gt; and Y&lt;table name&gt; are not processed by the WRITE method. (If no changes have been made, the processing can be skipped to save time.)</li> </ul> </li> </ul>

#### Export Parameter

Parameter Name	Value Help
CHANGENUMBER	Change number of the document

#### ↔ Sample Code

```
" example for change document object ZTEST_1
" with tables
" ZTEST_1 single case and
```

```

" ZTEST_2 multiple case
" Start of default parameter part
DATA: objectid      TYPE if_chdo_object_tools_rel=>ty_cdobjectv,
      utime        TYPE if_chdo_object_tools_rel=>ty_cduseit,
      udate        TYPE if_chdo_object_tools_rel=>ty_cddatum,
      username     TYPE if_chdo_object_tools_rel=>ty_cdusername,
      cdoc_upd_object TYPE if_chdo_object_tools_rel=>ty_cdchngindh VALUE
'U'.
DATA: cdchangenumber  TYPE if_chdo_object_tools_rel=>ty_cdchangenr.
" End of default parameter part
" Begin of dynamic DATA part for class ZCL_ZTEST_1_CHDO
" declaration for the long text : ICDTXT_ZTEST_1
DATA icdtxxt_ztest_1 TYPE if_chdo_object_tools_rel=>ty_cdtxt_tab.
" update indicator for the long text
DATA upd_icdtxxt_ztest_1 TYPE if_chdo_object_tools_rel=>ty_cdchngindh.
" workaera_old of ZTABLE_1
DATA os_ztable_1 TYPE ztable_1.
" workaera_new of ZTABLE1
DATA ns_ztable_1 TYPE ztable_1.
" change indicator for ZTABLE_1
DATA upd_ztable_1 TYPE if_chdo_object_tools_rel=>ty_cdchngindh.
" table with the NEW content of: ZTABLE_2
DATA xztable_2 TYPE zcl_ztest_1_chdo=>tt_ztable_2.
" table with the OLD content of: ZTABLE_2
DATA yztable_2 TYPE zcl_ztest_1_chdo=>tt_ztable_2.
" change indicator for table: ZTABLE_2
DATA upd_ztable_2 TYPE if_chdo_object_tools_rel=>ty_cdchngindh.
" Change Number of Document
DATA changenumber TYPE if_chdo_object_tools_rel=>ty_cdchangenr.
* save old values before change
* do some application changes to ZTABLE_1 and ZTABLE_2.
" set the change indicator for tables ('U', 'D', 'I')
" upd_ztable_1 = 'U'.
" upd_ztable_2 = 'U'.
" set change document application object identifier (objectid, date, time and
user)
" objectid          = 'Testobject'.
" utime            = 'hhmmss'.
" udate            = 'ddmmyyyy'.
" username         = 'Testusername'.
" set change indicator for change document header
cdoc_upd_object = 'U'.
" prepare old and new tables
SORT yztable_2.
DELETE ADJACENT DUPLICATES FROM yztable_2.
SORT xztable_2.
DELETE ADJACENT DUPLICATES FROM xztable_2.
" Begin of method call part
" define needed DATA for error handling
DATA err_ref TYPE REF TO cx_chdo_write_error.
DATA err_action TYPE string.
TRY.
  CALL METHOD zcl_ztest_1_chdo=>write
    EXPORTING
" Begin of default method call part
  objectid = objectid
  utime = utime
  udate = udate
  username = username
  object_change_indicator = cdoc_upd_object
" End of default method call part
" Begin of dynamic part for method call
  " declaration for the long text : ICDTXT_ZTEST_1
  icdtxxt_ztest_1 = icdtxxt_ztest_1
  " update indicator for the long text
  upd_icdtxxt_ztest_1 = upd_icdtxxt_ztest_1
  " workaera_old of ZTABLE_1
  o_ztable_1 = os_ztable_1

```

```

    " workaera_new of ZTABLE_1
    n_ztable_1 = ns_ztable_1
    " change indicator for ZTABLE_1
    upd_ztable_1 = upd_ztable_1
    " table with the NEW content of: ZTABLE_2
    xztable_2 = xztable_2
    " table with the OLD content of: ZTABLE_2
    yztable_2 = yztable_2
    " change indicator for table: ZTABLE_2
    upd_ztable_2 = upd_ztable_2
  " End of dynamic part for method call
  " Change Number of Document
  IMPORTING
    changenumber = changenumber.
  CATCH cx_chdo_write_error INTO err_ref.
    out->write( |Exception occurred: { err_ref->get_text( ) }| ).
ENDTRY.
...

```

#### 4.2.11.2.4 Deleting Change Documents

Use class `CL_CHDO_DELETE_TOOLS` to delete change documents.

First use `GET_INSTANCE` method to create an instance for the interface `IF_CHDO_DELETE_TOOLS`.

Parameter Name	Field Name	Value Help
RO_INSTANCE		Return parameter for instance creation
Method <code>IF_CHDO_DELETE_TOOLS~CHANGEDOCUMENT_DELETE</code> deletes the change documents for one change document object. You can restrict the selection by various parameters (such as date, change number or time).		
Import Parameters		
Parameter Name	Field Name	Value Help
I_OBJECTCLASS		Name of change document object
I_OBJECTID		Parameter of application objects IDs
I_UP_TO_DATE		Date by which the change documents are to be deleted
		All change documents until the specified date are deleted.
I_CHANGENUMBER		Change Number of Document
		If only one specific change document number is to be deleted, it can be transferred here.

Parameter Name	Field Name	Value Help
I_WITH_COMMIT		<p>Flag whether there should be a COMMIT WORK in the method.</p> <p>The flag determines whether a COMMIT WORK should be called in the method. This can be necessary when a lot is to be deleted. In this case the delete operations are divided up to avoid Roll-back segment overflow. The division is by clusters, so that no inconsistencies occur in the case of cancellation.</p>
COMMIT_COUNTER		<p>The parameter is only effective when a COMMIT WORK should be performed in the method (see parameter I_WITH_COMMIT).</p> <p>In the method, as many object change documents are read as are passed in COMMIT_COUNTER. These change documents are then deleted, and COMMIT WORK called. This procedure is repeated until there are no more change documents for the object before the until date.</p>

#### Export Parameter

Parameter Name	Field Name	Value Help
E_NUMBER_OF_DELETED_HEADERS		<p>Number of deleted change document headers</p> <p>This parameter contains the number of deleted change document headers</p>
E_NUMBER_OF_DELETED_POSITIONS		<p>Number of deleted change document items</p> <p>This parameter contains the number of deleted change document items.</p>
E_NUMBER_OF_DELETED_UIDS		<p>Number of deleted change document item GUIDs</p>
E_NUMBER_OF_DELETED_STRINGS		<p>Number of deleted change document item STRINGS</p> <p>This parameter contains the number of deleted change document items with STRINGS.</p>

## ↔ Sample Code

```
DATA lv_objectclass    TYPE if_chdo_object_tools_rel=>ty_cdobjectcl VALUE
'ZTEST_CO'.
  DATA lv_objectid     TYPE cl_chdo_write_tools=>ty_cdobjectv VALUE '9990001'.
  DATA lv_changenr     TYPE cl_chdo_write_tools=>ty_cdchangenr VALUE
'0000000001'.
  DATA lr_err          TYPE REF TO cx_chdo_delete_error.
).
TRY.
  DATA(lo_instance) = cl_chdo_delete_tools=>get_instance( ).
  lo_instance->if_chdo_delete_tools~changedocument_delete(
    EXPORTING
      i_objectclass           = lv_objectclass
      it_objectid             = lv_objectid
      i_changenumber          = lv_changenr
      i_with_commit            = abap_true
    IMPORTING
      e_number_of_deleted_headers = DATA(ls_headers)
      e_number_of_deleted_positions = DATA(ls_positions)
      e_number_of_deleted_uuids   = DATA(ls_uuids)
      e_number_of_deleted_strings = DATA(ls_strings)
    ).
  CATCH cx_chdo_delete_error INTO lr_err.
*   error handling for deletion
ENDTRY.
```

### 4.2.11.2.5 Reading Change Documents

Use class CL\_CHDO\_READ\_TOOLS to read change documents.

Method CHANGEDOCUMENT\_READ reads the change documents for one change document object. You can restrict the search by various parameters (such as changed by, date, or time).

Import Parameters

Parameter Name	Field Name	Value Help
ET_CDREDADD_TAB		Table type for structure CDREDADD, change documents return table
I_OBJECTCLAS		Name of change document object
IT_OBJECTID		Range table of application object IDs
I_DATE_OF_CHANGE		From-change date for search. All change documents are selected written on the specified date or later are found.
I_TIME_OF_CHANGE		From-change time for search. If no change time is specified, a selection is made from the time '000000'.

Parameter Name	Field Name	Value Help
I_DATE_UNTIL		Change date up to which you want to search. All change documents are selected written up to and including this date.
I_TIME_UNTIL		Latest change time in search. Time to which change documents are read on the "To" change date. If no time is passed, all change documents on the "To" change date are read.
IT_USERNAME		Username of the person responsible in change document. Only those change documents are selected that document changes made by this user. If no user name is passed, change documents are read for all users.
IS_READ_OPTIONS		
	local_time	If it is set, the date and time information in the formatted change documents is displayed in the local time of the user.
	time_zone	<p>It contains the time zone in which the change documents were written. If it's not set UTC applies.</p> <p>If it contains a time zone, this zone is used as the time zone in which the change documents were saved.</p> <p>If the change documents were saved in CET, the parameter must be set to CET.</p>

Parameter Name	Field Name	Value Help
	it_changenr	<p>Range table for change document number. Change document numbers were created internally as part of key of change documents. The key of change document is represented by Object name, Object ID of the application object and a change number. During creation of change documents using the write method of class &lt;name space&gt;CL_&lt;change document object name&gt;_CHDO</p> <p>Change documents numbers were received by export parameter CHANGENUMBER.</p>
	only_headers	Only return the change document header information without the position
IV_READ_ARCHIVE		Control the interface to read change documents from the archive, too

#### Export Parameter

Parameter Name	Field Name	Value Help
	OBJECTCLAS	Name of Change Document Object
	OBJECTID	Object ID of application object
	OBJECTID_DB	Object value
	CHANGENR	Change Number of Document
	OBJECTTXT	Object Description
	USERNAME	Username of the person responsible in change document
	USERNAME_DB	Username of the person responsible in change document
	UDATE	Creation date of the change document
	UDATE_DB	Creation date of the change document
	UTIME	Time changed
	UTIME_DB	Time changed

Parameter Name	Field Name	Value Help
	TCODE	Transaction in which a change was made
	APPLNAME	Application Object
	APPLTYPE	Application Type
	TABNAME	Change document creation: Table name
	TABNAME_DB	Change document creation: Table name
	TABKEY	Key of Changed Table Line
	TABKEY_DB	Key of Changed Table Line
	KEYLEN	Table key length
	CHNGIND	Type of Change
	FNAME	Field Name
	FNAME_DB	Field Name
	FTEXT	Explanatory Short Text
	TEXTART	Create change document: Text type
	SPRACHE	Language Key
	TEXT_CASE	Text change flag ('X')
	OUTLEN	Output length of the old and new value
	F_OLD	Old contents of changed field
	F_NEW	New contents of changed field
	F_NEW_DB	New contents of changed field
	VALUE_OLD	Old Extended Value (Long)
	VALUE_OLD_DB	Old Extended Value (Long)
	VALUE_NEW	New Extended Value (Long)
	VALUE_NEW_DB	New Extended Value (Long)
	VALUE_RAWSTR_OLD	Old Change Document Value for RAW-STRING Variable

Parameter Name	Field Name	Value Help
	VALUE_RAWSTR_OLD_DB	Old Change Document Value for RAW-STRING Variable
	VALUE_RAWSTR_NEW	New Change Document Value for RAW-STRING Variable
	VALUE_RAWSTR_NEW_DB	New Change Document Value for RAW-STRING Variable
	VALUE_SHSTR_OLD	Old Extended Value (Short)
	VALUE_SHSTR_OLD_DB	Old Extended Value (Short)
	VALUE_SHSTR_NEW	New Extended Value (Short)
	VALUE_SHSTR_NEW_DB	New Extended Value (Short)
	KEYGUID	KEYGUID for Link to CDPOS_UID
	TABKEY254	Key of Modified Table Row
	TABKEY254_DB	Key of Modified Table Row
	EXT_KEYLEN	Table key length
	KEYGUID_STR	KEYGUID for Link to CDPOS_STR
	VERSION	3-Byte field

## ↔ Sample Code

Read all change documents for object class ZCHDO\_TEST

```

CLASS zcl_chdo_read DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_read IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: rt_cdredadd TYPE cl_chdo_read_tools=>TT_CDREDADD_TAB,
          lr_err      TYPE REF TO cx_chdo_read_error.
    TRY.
      cl_chdo_read_tools=>changedocument_read(
        EXPORTING
          i_objectclass      = 'ZCHDO_TEST'    " change document object name
        *      it_objectid      =
        *      i_date_of_change   =
        *      i_time_of_change   =
        *      i_date_until       =
        *      i_time_until       =
        *      it_username        =

```

```

*      it_read_options  =
IMPORTING
      et_cdredadd_tab  = rt_cdredadd      " result returned in table
).
CATCH cx_chdo_read_error into lr_err.
  out->write( |Exception occurred: { lr_err->get_text( ) }| ).
ENDTRY.
ENDMETHOD.
ENDCLASS.

```

## ↳ Sample Code

Read all change documents for object class ZCHDO\_TEST including the archive.

```

CLASS zcl_chdo_read DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_chdo_read IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: rt_cdredadd TYPE cl_chdo_read_tools=>tt_cdredadd_tab,
          lr_err      TYPE REF TO cx_chdo_read_error.
    DATA lt_cdredadd TYPE cl_chdo_read_tools=>tt_cdredadd_tab.
    TRY.
      DATA(lt_files) =
        cl_arch_read_api=>get_files_to_read( iv_archiving_object = 'ZAOBJ_TEST' ).
        SORT lt_files BY creation_date DESCENDING creation_time DESCENDING.
        READ TABLE lt_files ASSIGNING FIELD-SYMBOL(<ls_file>) INDEX 1.
        CHECK sy-subrc = 0. " no files
        DATA(lo_read) = cl_arch_read_api=>get_instance( iv_archiving_object =
'ZAOBJ_TEST'
                                                iv_archive_key =
<ls_file>-archive_key ).
        DO.
          lo_read->get_next_data_object( IMPORTING ev_end_of_file =
DATA(lv_end_of_file)
                                         ev_archive_key =
DATA(lv_archive_key)
                                         ev_object_offset =
DATA(lv_offset) ).
            IF lv_end_of_file = abap_true.
              EXIT.
            ENDIF.
*             no application data
*             lo_read->get_data_records( EXPORTING iv_record_structure = 'XXX'
*                                         IMPORTING et_data_records = lt_data ).
*             APPEND LINES OF lt_data TO lt_data_all.
        TRY.
          cl_chdo_read_tools=>changedocument_read(
            EXPORTING
              i_objectclass      = 'ZCHDO_TEST'
              iv_read_archive   = lo_read
            IMPORTING
              et_cdredadd_tab  = DATA(lt_cd)
).
          CATCH cx_chdo_read_error INTO DATA(ls_read_err).
            out->write( |Exception occurred: { ls_read_err->get_text( ) }| ).
        ENDTRY.
        APPEND LINES OF lt_cd TO lt_cdredadd.
      ENDDO.

```

```
    lo_read->close( ).  
    CATCH cx_arch_api INTO DATA(lx_error).  
        out->write( |Exception occurred: { lx_error->get_text( ) }| ).  
    ENDTRY.  
ENDMETHOD.
```

## 4.2.11.2.6 Archiving Change Documents

You can archive change documents together with their application documents.

### Prerequisites

1. A change document object is available (see [Working with Change Document Objects](#)) and it's implemented to create change documents (see [Writing Change Documents \[page 1448\]](#)).
2. There is an archiving object to archive current application documents (see [Working with Archiving Objects](#)).

### Procedure

1. Add the archiving class CHANGEDOCU to the archiving classes of your archiving object using ABAP Development Tools (ADT):

## Archiving Object: CDTESTLV5

### Attributes

<u>Write Class:</u> *	CL_CDTESTLV5_WRI
<u>Delete Class:</u> *	CL_CDTESTLV5_DEL
<u>Reload Class:</u>	CL_CDTESTLV5_REL
<u>Storage Class:</u> *	CL_CDTESTLV5_STORAGE_MANAGER

### Tables to be Archived

Name

### Archiving Classes

Name
CHANGEDOCU

- Update the write class of your archiving object to archive change documents.

### Sample Code

```
CLASS ZCL_<YOUR_ARCH_OBJECT>_WRI IMPLEMENTATION.
METHOD if_apj_rt_exec_object~execute.
...
TRY.
DATA(lo_write) = cl_arch_write_api->get_instance( iv_archiving_object =
<YOUR_ARCH_OBJECT> iv_testmode = lv_test ).
LOOP AT lt_keys_tab ASSIGNING FIELD-SYMBOL(<fs_keys>).
lo_write->open_data_object( ).
cl_changedocu_archive=>get_instance( )-
>if_changedocu_archive~changedocu_archive_object( iv_objectclass
= <YOUR_CD_OBJECTCLASS>
iv_objectid = <fs_keys>-objectid
iv_write_instance = lo_write ).
lo_write->close_data_object( ).
ENDLOOP.
lo_write->finalize( ).
CATCH cx_arch_api INTO DATA(lx_error).
CATCH cx_chdo_archive_error INTO DATA(lx_error2).
ENDTRY.
ENDMETHOD.
```

#### ⓘ Note

String <YOUR\_ARC\_H\_OBJECT> is for your archiving object. String <YOUR\_CD\_OBJECTCLASS> is for your change document object.

#### ⓘ Note

To read change documents from the archive, consider [Reading Change Documents \[page 1455\]](#).

## 4.2.11.2.7 How to Use the Fiori Reuse Library for Change Document Solution

You can use the *Reuse Library* to implement a reusable component that can display change documents.

### Context

To be able to use the reusable components, you need to adapt the code that can be found in your `manifest.json` file.

#### ⚠ Restriction

Mind that the Fiori reuse library is only released for the OData V2 protocol.

### Procedure

1. Adapt your `<manifest.json>` file and add the `sap.nw.core.changedocs.lib.reuse` library under the `sap.ui5` section:

#### ↔ Sample Code

```
"sap.ui5": {
    "resources": {
        "js": [],
        "css": []
    },
    "componentUsages": {
        "ChangedocReuseComponent": {
            "name": "sap.nw.core.changedocs.lib.reuse.changedocscomponent"
        }
    },
    "dependencies": {
        "minUI5Version": "1.65.0",
        "libs": {
            "sap.ui.core": {
                "lazy": false
            }
        }
    }
}
```

```

        },
        "sap.ui.generic.app": {
            "lazy": false
        },
        "sap.suite.ui.generic.template": {
            "lazy": false
        },
        "sap.ui.comp": {
            "lazy": false
        }
    },
    "components": {
        "sap.nw.core.changedocs.lib.reuse": {
            "lazy": true
        }
    }
},
}

```

2. Adapt your <manifest.json> file and add the component as section on your page:

#### ↔ Sample Code

```

"pages": {
    "ObjectPage|TravelProcessor": {
        "entitySet": "TravelProcessor",
        "defaultLayoutTypeIfExternalNavigation": "MidColumnFullScreen",
        "component": {
            "name": "sap.suite.ui.generic.template.ObjectPage"
        },
        "embeddedComponents": {
            "changedoccomponent": {
                "id": "changedoccomponent",
                "componentUsage": "ChangedocReuseComponent",
                "title": "{reuseTitle}",
                "settings": {
                    "objectClass": [
                        "<MY_CHANGE_DOC>",
                        ],
                    "objectId": "{parts:
[ {path: '<MY_OBJECTID>' }], formatter:
'sap.nw.core.changedocs.lib.reuse.changedocscomponent.arrayFormatter'}",
                    "startDate": "1900-01-01T00:00:00"
                }
            }
        }
    }
}

```

#### ⓘ Note

Make sure to replace strings <MY\_CHANGE\_DOC> with your own change document object and <MY\_OBJECTID> with your own Object id which were also used to create change documents.

3. Adapt your <i18n.properties> file and add:

#### ↔ Sample Code

```

...
#XTIT: Reuse component title

```

```
reuseTitle =Change Documents  
...
```

### ⓘ Note

Manual tests of the reuse library using *SAP Business Application Studio* is not possible. Create and test your app finally before implementing the reuse app. Test the reuse app after deployment.

After your app was deployed successfully to an SAP Business Technology Platform system, the updated BSP application and the SAP Fiori Launchpad app descriptor item will appear under your created package in Eclipse.

4. Update your IAM App. For more information, see [Defining an IAM App for the Business Service \[page 826\]](#).
5. Once created, maintain the *Change Document OData Service*. Go to the Service tab and add the *Change Document Log OData Service* by naming the *OData V2* service type and add the following service name: **APS\_CHANGE\_DOCUMENTS\_SRV 0001** (**Note: make sure to add all 11 spaces in between 'SRV' and '0001'**).
6. Create a new Business Catalog. For more information, see [Creating a Business Catalog \[page 827\]](#)

Having created the business catalog, you have successfully implemented a reusable component to display your change documents.

## 4.2.11.3 Currency Conversion

The ABAP Core Data Services (CDS) provide the built-in function `CURRENCY_CONVERSION` to convert different currencies to a common target currency.

For more information, see [CDS DDL - CDS View Entity, Unit and Currency Conversion Functions](#) in the [ABAP Keyword Documentation](#).

To customize this conversion function, you can use API class `CL_EXCHANGE_RATES` to load the current exchange rates into your system for a correct conversion. Class `CL_EXCHANGE_RATES` is based on the business API function modules `BAPI_EXCHRATE_CREATEMULTIPLE` and `BAPI_EXCHANGERATE_SAVEREPLICA`.

API class `CL_EXCHANGE_RATES` provides PUT methods to write exchange rates to the corresponding customizing persistence. For more information, see the ABAP Doc comments in the class implementation.

Note that aside from use in CDS, the class additionally provides two methods to convert currencies.

### Example

To learn how to obtain and store exchange rates from an official source, see the detailed example at <https://github.com/SAP-samples/cloud-abap-exchange-rates>.

#### ↗ Sample Code

```
@EndUserText.label: 'Price (in US American Dollars)'  
currency_conversion
```

```

client => client,
amount => amount,
round => '',
source_currency => currency,
target_currency => cast('USD' as abap.cuky),
exchange_rate_type => cast('M' as abap.char(4)),
exchange_rate_date => cast($session.system_date as abap.dats)
) as PriceInUSD

```

## 4.2.11.4 Factory Calendar

Get an overview of the ABAP classes, interfaces, and methods you can use to access calendar-related information.

The calendar system comprises the definition of the following categories:

- Public holidays
- Holiday calendars (a collection of public holidays)
- Factory calendars

A factory calendar is based on a country-specific holiday calendar and defines which days are working days, for example, Monday to Friday, and location-specific rules.

Read on to learn about the set of data we deliver for these categories.

### Note

Class `CL_SCAL_API` is still available, but deprecated. We recommend that you use class `CL_FHC_CALENDAR_RUNTIME` and the related interfaces `IF_FHC_FCAL_RUNTIME`, `IF_FHC_HCAL_RUNTIME` and `IF_FHC_HOLIDAY_RUNTIME` instead. Furthermore, the CDS views from the section [Accessing Holiday and Factory Calendar Tables Using VDM-Compliant CDS Views \[page 1470\]](#) can be used to get an overview of existing factory- and holiday calendars. For information about the mapping between SCAL-relevant IDs and FHC-relevant IDs, see the corresponding section below.

## Creating a Runtime to Access Calendar-Related Information

Use class `CL_FHC_CALENDAR_RUNTIME` that contains the following list of methods to create a runtime for holidays, holiday calendars, and factory calendar:

Method	Description
<code>create_factorycalendar_runtime</code>	Provides a factory calendar runtime
<code>create_holidaycalendar_runtime</code>	Provides a holiday calendar runtime
<code>create_holiday_runtime</code>	Provides a holiday runtime

## ↔ Sample Code

```
try.  
    data(lo_fcal_run) =  
        cl_fhc_calendar_runtime->create_factorycalendar_runtime (  
            iv_factorycalendar_id = 'ExampleID' ).  
    catch cx_fhc_runtime into data(lx_err).  
        "exception handling  
endtry.
```

## Reading Factory Calendar-Related Information

Use the before created runtime to access the following list of methods provided in the interface `IF_FHC_FCAL_RUNTIME` to get information about the factory calendar:

Method	Description
convert_date_to_factorydate	Converts a date to a factory date
convert_factorydate_to_date	Converts a factory date to a date
get_last_factorydate	Provides the last factory date of the calendar
calc_workingdays_between_dates	Calculates the number of working days between two dates
add_workingdays_to_date	Adds working days to a date
subtract_workingdays_from_date	Subtracts working days to a date
get_validity_start	Provides the first valid date
get_validity_end	Provides the last valid date
is_date_workingday	Check if a weekday is a working day
is_holiday_workingday	Check if holiday is a working day
get_description	Provides the description of the calendar
get_hcal_assignment	Provides the assigned holiday calendar runtime
get_id	Provides the ID of the factory calendar

## ↔ Sample Code

```
try.  
    data(lv_date) = lo_fcal_run->convert_factorydate_to_date(  
        iv_factorydate = '123' ).  
    catch cx_fhc_runtime into data(lx_run_err).  
        "exception handling  
endtry.
```

## Reading Holiday Calendar-Related Information

Use the before created runtime to access the following list of methods provided in the interface `IF_FHC_HCAL_RUNTIME` to get information about the holiday calendar:

Method	Description
<code>is_holiday</code>	Check if a date is a holiday
<code>get_holiday</code>	Provides the assigned holiday for a date
<code>calc_holidays_between_dates</code>	Calculates the number of holidays between two dates
<code>get_validity_start</code>	Provides the first valid date
<code>get_validity_end</code>	Provides the last valid date
<code>get_description</code>	Provides the description of the calendar
<code>get_holiday_assignments</code>	Get the assigned holidays from current holiday calendar
<code>get_id</code>	Provides the ID of the holiday calendar

## Reading Holiday-Related Information

Use the before created runtime to access the following list of methods provided in the interface `IF_FHC_HOLIDAY_RUNTIME` to get information about the holidays:

Method	Description
<code>get_holiday_id</code>	Provides the ID of the holiday
<code>get_type</code>	Provides the type of the holiday
<code>get_class</code>	Provides the class of the holiday
<code>get_confession</code>	Provides the confession of the holiday
<code>get_text</code>	Provides the title and description of the holiday in a specific language

## Mapping between SCAL-relevant IDs and FHC-relevant IDs

Class `CL_FHC_CALENDAR_RUNTIME` uses longer IDs (32 digits) than the deprecated class `CL_SCAL_API`, which had only two-digit IDs for the factory calendars and holiday calendars, and three-digit IDs for the public holidays (also referred to as legacy IDs). The longer IDs make it easier to categorize calendar data.

To enable the use of the above mentioned runtimes, it may be necessary that an application establishes a connection between a legacy two-digit ID (as formerly used with CL\_SCAL\_API) and a longer ID that the newer class CL\_FHC\_CALENDAR\_RUNTIME uses.

You can do such a mapping using the interface IF\_FHC\_ID\_MAPPER and class CL\_FHC\_CALENDAR\_ID\_MAPPER. The mapping is available for holidays, holiday calendars, and factory calendars.

The CL\_FHC\_CALENDAR\_ID\_MAPPER class returns an ID mapper instance. Subsequently, you can use the methods provided by the interface IF\_FHC\_ID\_MAPPER to perform the desired mappings.

#### ⚠ Caution

It can't be guaranteed that the 2-digit legacy ID will point to the same calendar on each system. The 2-digit legacy ID should only be used if applications have already used CL\_SCAL\_API, stored the 2-digit legacy ID and have not yet migrated to the new APIs and IDs. The 2-digit legacy IDs are only intended for the transition phase.

We recommend using and saving the new 32-digit IDs on the application side. The whole functionality is optimized for using the new 32-digit ID.

### Creating an ID Mapper Instance

Method	Description
create_id_mapper	Provides an ID mapper instance

#### ↔ Sample Code

```
data(lo_id_mapper) = cl_fhc_calendar_id_mapper->create_id_mapper( ).
```

### Mapping the IDs

Method	Description
mapping_fcal_legacyid_to_id	Provides the 32-digit factory calendar ID for a two-digit SCAL factory calendar ID
mapping_hcal_legacyid_to_id	Provides the 32-digit holiday calendar ID for a two-digit SCAL holiday calendar ID
mapping_hol_legacyid_to_id	Provides the 32-digit holiday ID for a three-digit SCAL holiday ID
mapping_fcal_id_to_legacyid	Provides the two-digit SCAL factory calendar ID for a 32-digit factory calendar ID
mapping_hcal_id_to_legacyid	Provides the two-digit SCAL holiday calendar ID for a 32-digit holiday calendar ID
mapping_hol_id_to_legacyid	Provides the three-digit SCAL holiday calendar ID for a 32-digit holiday ID

#### ↔ Sample Code

```
try.
```

```

        data(lv_new_id) = lo_id_mapper->mapping_fcal_legacyid_to_id( iv_legacy_id
= '01' ).
        catch cx_fhc_runtime into data(lx_err).
        "exception handling
endtry.

"another example

try.
    data(lv_legacy_id) = lo_id_mapper-
>mapping_fcal_id_to_legacyid( iv_factorycalendar_id = 'ExampleID' ).
    catch cx_fhc_runtime into lx_err.
    "exception handling
endtry.

```

## Reading Additional Factory Calendar-Related Information

Independently from the above factory calendar information, you can use class `CL_SCAL_UTILS`, which contains the following methods:

Method	Description
MONTH_NAMES_GET	Provides the names of the months
WEEK_GET_FIRST_DAY	Provides the first day of a week
DATE_GET_WEEK	Provides the year and week of a date
DATE_COMPUTE_DAY	Provides the name and number of the weekday for a specified date

### ↳ Sample Code

```

try.
    cl_scal_utils=>date_get_week(
        exporting
            iv_date = '20201001'
        importing
            ev_year = data(lv_year)
            ev_week = data(lv_week) .
    catch cx_scal into data(lx_scal).
    "exception handling
endtry.

```

### 4.2.11.4.1 Accessing Holiday and Factory Calendar Tables Using VDM-Compliant CDS Views

You can access the factory calendar and holiday calendar tables using VDM-compliant CDS views.

The VDM-compliant CDS views used here are CDS views that were created in `SAP_BASIS` and belong to the component `BC-SRV-ASF-CAL`.

The following CDS views access the factory- and holiday calendar tables:

CDS View	Description
I_PublicHolidayCalendarBasic	General information about holiday calendars
I_PublHolidayCalendarBasicText	Maintained description for holiday calendars
I_PublicHolidayCalendarVH	Value help for holiday calendars containing the holiday calendar ID, legacy holiday calendar ID, the validity start and end date and description
I_FactoryCalendarBasic	General information about a factory calendar
I_FactoryCalendarBasicText	Maintained description for factory calendars
I_FactoryCalendarValueHelp	Value help for factory calendars containing the factory calendar ID, legacy factory calendar ID, the validity start and end date and description

#### ⓘ Note

The CDS views mentioned here should be differentiated from the C1-released VDM CDS views created by SAP S/4HANA. An example for a C1-released CDS view is `I_FactoryCalendar`. C1-released VDM CDS views refer to the tables of the factory- and holiday calendar used previously. They don't belong to component BC-SRV-ASF-CAL. For this reason, the name addition `BASIC` was introduced for the new version of the CDS views, as can be seen below.

## 4.2.11.5 Lock Objects

When you activate a lock object in the ABAP Dictionary, the system automatically creates function modules for setting locks (`ENQUEUE_<lock object name>`) and releasing locks (`DEQUEUE_<lock object name>`).

### Using Lock Objects

Since the direct usage of these function modules is not permitted, the function module calls are wrapped in a generic API to set and release locks.

- Use class `CL_ABAP_LOCK_OBJECT_FACTORY` to do the following:
  - Create a lock object instance using method `GET_INSTANCE`
  - Release all locks of the current LUW using method `DEQUEUE_ALL`
- Use the created lock object instance to do the following:
  - Set a lock using method `ENQUEUE`
  - Release a lock using method `DEQUEUE`

For more information, see the ABAP Doc comments of class `CL_ABAP_LOCK_OBJECT_FACTORY` and interface `IF_ABAP_LOCK_OBJECT`. The interface `IF_ABAP_LOCK_OBJECT` also contains constants to be used for parameterization of lock methods.

#### ↴ Sample Code

```
data lv_value type <type of lock object parameter 1>.
```

```

lv_value = <value for lock parameter 1>.
data(lr_lock_object) = cl_abap_lock_object_factory->get_instance(
    exporting iv_name = <name of lock object> ).
try.
    lr_lock_object->enqueue(
        it_parameter = value #( ( name = <name of lock parameter 1> value =
ref #( lv_value ) ) )
        it_table_mode = value #( ( table_name = <table name> mode =
if_abap_lock_object=>cs_mode-<constant for lock mode>) ) .
    catch cx_abap_lock_failure into data(lr_exc).
        <exception handling code>
    endtry.
<application code>
try.
    lr_lock_object->dequeue( ).
    catch cx_abap_lock_failure into lr_exc.
        <exception handling code>
    endtry.

```

## 4.2.11.6 Number Range Solution

Many business applications require unique numbers, for example, to complete the keys of data records. In order to get numbers from an interval, a number range object must be defined which can contain different properties. In addition, intervals containing the numbers, must be assigned to the number range object. Numbers can be generated from existing number range intervals.

### ⓘ Note

Creation, change, and deletion of number range objects and intervals require developer role authorization. Changes to objects and intervals can only be performed in the same software layer.

### 4.2.11.6.1 Maintaining Number Range Objects

To maintain a number range object during design time, the ABAP development tools for Eclipse UI editor for number range objects can be used. For more information, see [Working with Number Range Objects](#).

Class `CL_NUMBERRANGE_OBJECTS` provides methods for maintaining number range objects.

For more information, see

- [Creating Number Range Objects \[page 1473\]](#)
- [Changing Number Range Objects \[page 1475\]](#)
- [Deleting Number Range Objects \[page 1476\]](#)
- [Reading Number Range Objects \[page 1476\]](#)

## Related Information

[Working with Number Range Objects](#)

### 4.2.11.6.1.1 Creating Number Range Objects

Use method CREATE to create number range objects.

When creating a number range object, the following naming-rules apply:

- For customer objects, the name must start with a 'Z' or a 'Y'.
- The maximum length of a number range object is 10 characters.

Import Parameters

Parameter Name	Field Name	Value Help
ATTRIBUTES		Number Range Object Definition.
	OBJECT	Name Range Object.
	DTELSOBJ	Data element for sub-object.
	YEARIND	Flag, whether number range object is to- year relevant.
	DOMLEN	Domain, which determines the length of the numbers.
	PERCENTAGE	Percentage of numbers remaining in an interval after having identified in which number assignment a warning is given.
	CODE	Transaction code to call interval maintenance (obsolete for ABAP CP).
	BUFFER	Buffering type for number assignment.
	NOIVBUFFER	Number of numbers in the buffer.
	NRSWAP	Selecting the flag prevents the intervals from automatically starting from the beginning at the upper limit.
	NRCHECKASCII	
	DEVCLASS	Development class of the object.
OBJ_TEXT	CORRNR	Correction number for transport.

Parameter Name	Field Name	Value Help
		Texts for objects for change document object creation.
	LANGU	Language.
	TXT	Description of object, long text.
	TXTSHORT	Description of object, short text.
Export Parameters		
Parameter Name	Field Name	Value Help
ERRORS		
	MSGID	Message class (NR)
	MSGTYPE	Message type
	MSGNUMBER	Message number
	MSGVAR1	Variable to message
	MSGVAR2	Variable to message
	MSGVAR3	Variable to message
	MSGVAR4	Variable to message
	TABLENAME	Table
	FIELDNAME	Field
	CRITCHANGE	Critical change
RETURNCODE		Space: no error E: error W: warning

### ↳ Sample Code

```

CLASS zcl_nr_object_create DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_nr_object_create IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA:
      lv_object    TYPE cl_numberrange_objects=>nr_attributes-object,

```

```

lv_devclass  TYPE cl_numberrange_objects=>nr_attributes-devclass,
lv_corrhnr   TYPE cl_numberrange_objects=>nr_attributes-corrhnr.
lv_object    = 'Z_TEST_03'.
lv_devclass  = 'Z_SNUM'.
lv_corrhnr   = 'SIDK123456'.
TRY.
cl_numberrange_objects=>create(
  EXPORTING
    attributes = VALUE #( object      = lv_object
                           domlen      = 'CHAR8'
                           percentage  = 5
                           buffer      = abap_true
                           noivbuffer = 10
                           devclass   = lv_devclass
                           corrnrs    = lv_corrhnr )
    obj_text   = VALUE #( object      = lv_object
                           langu       = 'E'
                           txt         = 'Create object'
                           txtshort   = 'Test create' )
  IMPORTING
    errors      = DATA(lt_errors)
    returncode  = DATA(lv_returncode)
  ).
CATCH.
...
ENDTRY.
...
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.6.1.2 Changing Number Range Objects

Use method CHANGE to update number range objects.

Import and export parameters are the same as in the CREATE method.

### Sample Code

```

...
lv_object    = 'Z_TEST_03'.
lv_devclass  = 'Z_SNUM'.
lv_corrhnr   = 'SIDK123456'.
...
cl_numberrange_objects=>update(
  EXPORTING
    attributes = VALUE #( object      = lv_object
                           domlen      = 'CHAR8'
                           percentage  = 9
                           buffer      = 'S'
                           noivbuffer = 12
                           devclass   = lv_devclass
                           corrnrs    = lv_corrhnr )
    obj_text   = VALUE #( object      = lv_object
                           langu       = 'E'
                           txt         = 'Update object'
                           txtshort   = 'Test update' )
  IMPORTING
    errors      = DATA(lt_errors)
    returncode  = DATA(lv_returncode)
  ).
...
```

### 4.2.11.6.1.3 Deleting Number Range Objects

Use method `DELETE` to delete number range objects.

Import Parameters

Parameter Name	Field Name	Value Help
OBJECT		Number Range Object
CORRNR		Correction number for transport

#### ↳ Sample Code

```
...
lv_object = 'Z_TEST_03'.
lv_corrnr = 'SIDK123456'.
...
cl_numberrange_objects=>delete(
  EXPORTING
    object = lv_object
    corrnr = lv_corrnr
  ).
...
...
```

### 4.2.11.6.1.4 Reading Number Range Objects

Use the `READ` method to read the attributes of a number range object.

Import Parameters

Parameter Name	Field Name	Value Help
LANGUAGE		Language for the object texts
OBJECT		Number Range Object

Export Parameters

Parameter Name	Field Name	Value Help
ATTRIBUTES		Number Range Object Definition.
	OBJECT	Number Range Object.
	DTELSOBJ	Data element for sub-object.

Parameter Name	Field Name	Value Help
	YEARIND	Flag, whether number range object is to-year relevant.
	DOMLEN	Domain, which determines the length of the numbers.
	PERCENTAGE	Percentage of numbers remaining in an interval after having identified in which number assignment a warning is given.
	CODE	Transaction code to call interval maintenance (obsolete for ABAP CP).
	BUFFER	Buffering type for number assignment.
	NOIVBUFFER	Number of numbers in the buffer.
	NRSWAP	Selecting the flag prevents intervals from automatically starting from the beginning at the upper limit.
	NRCHECKASCII	
INTERVAL_EXISTS		
OBJ_TEXT		Texts for objects for change document object creation.
	LANGU	Language.
	TXT	Description of object, long text.
	TXTSHORT	Description of object, short text.

### ↔ Sample Code

```

...
lv_object = 'Z_TEST_03'
...
cl_numberrange_objects->read(
    EXPORTING
        language      = sy-langu
        object        = lv_object
    IMPORTING
        attributes    = DATA(ls_attributes)
        interval_exists = DATA(lv_interval_exists)
        obj_text      = DATA(obj_text)
).
...

```

## 4.2.11.6.2 Maintaining Intervals of Number Range Objects

The class `CL_NUMBERRANGE_INTERVALS` provides methods for maintaining intervals of number range objects. Will the APIs be used by a business user, assign the `SAP_CA_BC_IC_LND_NUM_PC` (Number Range Management – Configuration) business catalog to a proper business role and - if required - restrict it to specific number range objects.

For more information, see

- [Creating Intervals of Number Range Objects \[page 1478\]](#)
- [Changing Intervals of Number Range Objects \[page 1480\]](#)
- [Deleting Intervals of Number Range Objects \[page 1481\]](#)
- [Reading Intervals of Number Range Objects \[page 1481\]](#)

### 4.2.11.6.2.1 Creating Intervals of Number Range Objects

Use the `CREATE` method to create number range intervals.

Import Parameters

Parameter Name	Field Name	Value Help
INTERVAL		Interval Table
	SUBOBJECT	Number Range Object Sub-object Value
	NRRANGENR	Number Range Number
	TOYEAR	To Fiscal Year
	FROMNUMBER	From Number
	TONUMBER	To Number
	NRLEVEL	Number Range Level
	EXTERNIND	Internal ('I') or external ('X') number range flag
	PROCIND	Processing flag (I=Insert, D=Delete, U=Update, '=no changes)
OBJECT		Number Range Object
SUBOBJECT		Sub-object

## Export Parameters

Parameter Name	Field Name	Value Help
ERROR		Flag showing that an error occurred during testing
ERROR_INF		Error Information
	MSGNR	Message Number
	TABLENAME	Parameter Name
	FIELDNAME	Field Name
	TABIX	Index of Row with Error
ERROR_IV		Intervals with errors
	SUBOBJECT	Number range object subobject value
	NRRANGENR	Number range number
	TOYEAR	To fiscal year
	FROMNUMBER	From number
	TONUMBER	To number
	NRLEVEL	Number range level
	EXTERNIND	Internal ('I') or external ('X') number range flag
	PROCIND	Processing flag (I=Insert, D=Delete, U=Update, '=no changes)
WARNING		Flag: Warning after check?

### ↳ Sample Code

```
...
CLASS zcl_nr_test_intervals_create DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_nr_test_intervals_create IMPLEMENTATION.
METHOD if_oo_adt_classrun~main.
DATA: lv_object      TYPE cl_numberrange_objects=>nr_attributes-object,
      lt_interval   TYPE cl_numberrange_intervals=>nr_interval,
      ls_interval   TYPE cl_numberrange_intervals=>nr_nriv_line.
*     lv_object = 'Z_TEST_03'.
*     intervals
```

```

ls_interval-nrrangernr = '01'.
ls_interval-fromnumber = '00000001'.
ls_interval-tonumber   = '19999999'.
ls_interval-procind    = 'I'.
APPEND ls_interval TO lt_interval.
ls_interval-nrrangernr = '02'.
ls_interval-fromnumber = '20000000'.
ls_interval-tonumber   = '29999999'.
APPEND ls_interval TO lt_interval.
* create intervals
TRY.
    out->write( |Create Intervals for Object: { lv_object } | ).
CALL METHOD cl_numberrange_intervals=>create
    EXPORTING
        interval = lt_interval
        object   = lv_object
        subobject = ''
    IMPORTING
        error     = DATA(lv_error)
        error_inf = DATA(ls_error)
        error_iv  = DATA(lt_error_iv)
        warning   = DATA(lv_warning).
    ENDTRY.
ENDMETHOD.
ENDCLASS.
...

```

## 4.2.11.6.2.2 Changing Intervals of Number Range Objects

Use the UPDATE method to change number range intervals.

Import and export parameters are the same as in the CREATE method.

### ↳ Sample Code

```

...
lv_object = 'Z_TEST_03'.
* intervals
ls_interval-nrrangernr = '01'.
ls_interval-fromnumber = '00000002'.
ls_interval-tonumber   = '19999998'.
ls_interval-procind    = 'U'.
APPEND ls_interval TO lt_interval.
ls_interval-nrrangernr = '02'.
ls_interval-fromnumber = '20000002'.
ls_interval-tonumber   = '29999997'.
APPEND ls_interval TO lt_interval.
...
CALL METHOD cl_numberrange_intervals=>update
    EXPORTING
        interval = lt_interval
        object   = lv_object
        subobject = ''
    IMPORTING
        error     = DATA(lv_error)
        error_inf = DATA(ls_error)
        error_iv  = DATA(lt_error_iv)
        warning   = DATA(lv_warning)
...

```

### 4.2.11.6.2.3 Deleting Intervals of Number Range Objects

Use the `DELETE` method to delete number range intervals.

Import and export parameters are the same as in the `CREATE` method.

#### ↳ Sample Code

```
...
*   lv_object = 'Z_TEST_03'
*   intervals
ls_interval-nrrangenr  = '01'.
ls_interval-fromnumber = '00000001'.
ls_interval-tonumber   = '19999999'.
ls_interval-procind    = 'D'.
APPEND ls_interval TO lt_interval.
ls_interval-nrrangenr  = '02'.
ls_interval-fromnumber = '20000000'.
ls_interval-tonumber   = '29999999'.
APPEND ls_interval TO lt_interval.

...
CALL METHOD cl_numberrange_intervals=>delete
  EXPORTING
    interval  = lt_interval
    object    = lv_object
    subobject = ''
  IMPORTING
    error      = DATA(lv_error)
    error_inf  = DATA(ls_error)
    error_iv   = DATA(lt_error_iv)
    warning    = DATA(lv_warning).
...
...
```

### 4.2.11.6.2.4 Reading Intervals of Number Range Objects

Use the `READ` method to get the properties of number range intervals.

Import Parameters

Parameter Name	Field Name	Value Help
NR_RANGE_NR1		Interval Number (internal interval)
NR_RANGE_NR2		Interval Number (external interval)
OBJECT		Number Range Object
SUBOBJECT		Sub-object

#### ↳ Sample Code

```
...
*   lv_object = 'Z_TEST_03'.
...
CALL METHOD cl_numberrange_intervals=>read
```

```
EXPORTING
    object      = lv_object
    nr_range_nr1 = ''
    nr_range_nr2 = ''
    subobject   = ''
IMPORTING
    interval    = lt_interval.
...

```

#### 4.2.11.6.2.5 How to Create a Fiori App for a single Number Range Object

You can maintain files in the SAP Fiori Launchpad to launch the Number Range app with the help of a custom app.

#### Prerequisites

The business user will need to be assigned with suitable authorization in the Steampunk system:

- the SAP\_CA\_BC\_IC\_LND\_NUM\_PC catalog is needed, as it grants access to the Maintain Number Ranges app
- the SAP\_CA\_BC\_IC\_LND\_PC catalog is needed, as it grants access to the customizing changes
- if the targeted business configuration instance is protected, the relevant custom business catalog must also be assigned to the business user

#### Procedure

A custom tile that navigates directly to the Number Range Object app has been created. This is done by using the parameter `Technical Identifier` in the intent-based navigation .

The implementation contains a creation and deployment of an empty freestyle UI5 app that redirects to the `NumberRangeInterval-manage` (manage number range intervals) with the required parameter. When clicking the tile, the launchpad will first navigate to the intent of the custom app (which is maintained in the app descriptor) and will then automatically be rerouted to the intent of the Maintain Business Configurations app. This rerouting is quite simple to implement after generating an empty app in Business Application Studio (BAS).

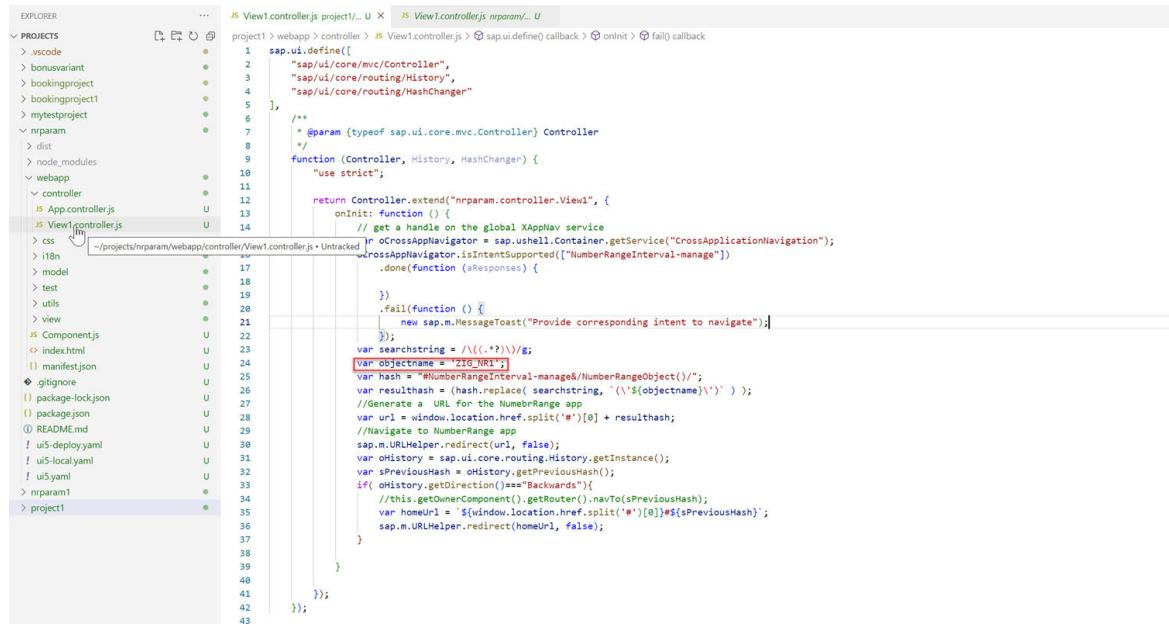
In terms of authorization control, a custom business catalog and business role needs to be created in the Steampunk system and assigned to the business user.

## Reproduce

A development package and a transport request are needed to implement the solution.

### Create a new SAP Fiori application in BAS

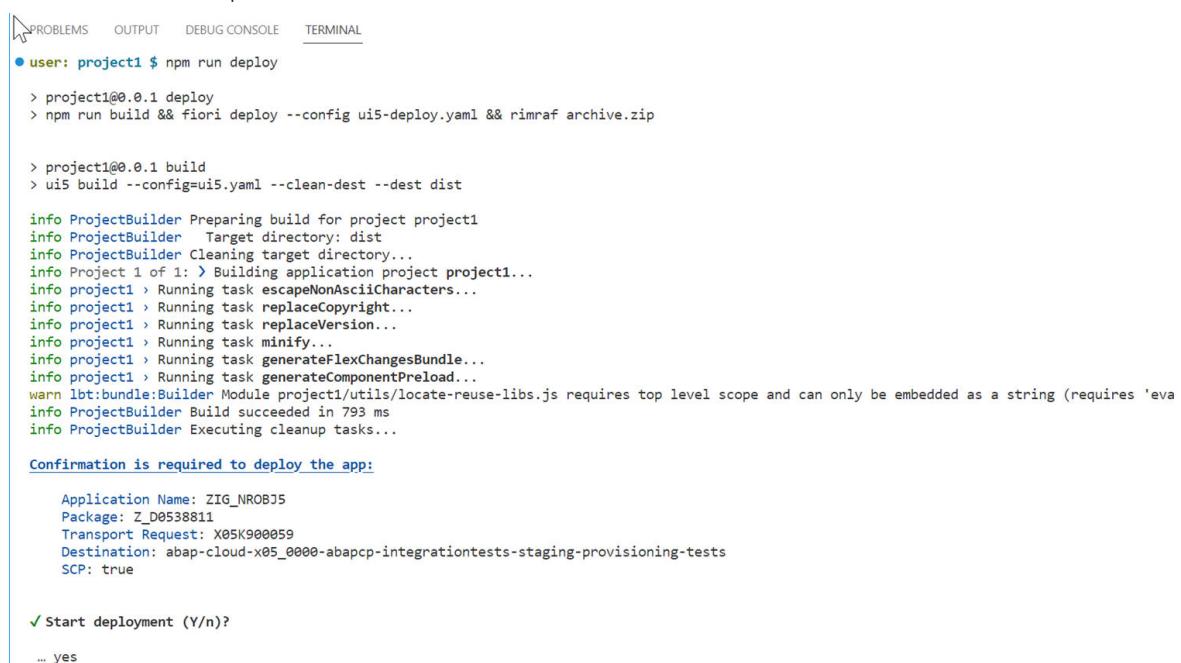
1. Log on to the Business Application Studio (BAS) Development Space to create a new SAP Fiori application in BAS and press *Start*.
2. Choose SAPUI5 freestyle from the drop-down menu *Application Type* and press *Next*.



The screenshot shows the SAP Business Application Studio interface with the code editor open. The file is `View1.controller.js` under project `project1`. The code defines a controller for a view named `View1`. It includes logic to handle navigation using the `XAppNav` service. A specific section of the code is highlighted, showing a regular expression search string and a variable definition:

```
var searchstring = '/(IG_NR1)/g';
var objectname = "IG_NR1";
var hash = "#NumberRangeInterval-manage&NumberRangeObject/";
var resulthash = (hash.replace(searchstring, ('${objectname}\')'));
//Generate a URL for the NumberRange app
var url = window.location.href.split('#')[0] + resulthash;
//Navigate to NumberRange app
sap.m.URLHelper.redirect(url, false);
var oCrossAppNavigator = sap.ui.core.routing.history.getInstance();
var sPreviousHash = oHistory.getPreviousHash();
if(oHistory.getNextDirection() == "Backwards"){
    //this.getOwnerComponent().getRouter().navTo(sPreviousHash);
    var homeurl = ${window.location.href.split('#')[0]}#${sPreviousHash};
    sap.m.URLHelper.redirect(homeurl, false);
}
```

3. For *Data source* choose *None* out of the drop-down menu and press *Next*. You can use the proposed view name in the next step or rename it. Press *Next*.



The screenshot shows the SAP Business Application Studio terminal output during the deployment process. The command `npm run deploy` is executed, and the output shows the build and deployment steps:

```
user: project1 $ npm run deploy

> project1@0.0.1 deploy
> npm run build && fiori deploy --config ui5-deploy.yaml && rimraf archive.zip

> project1@0.0.1 build
> uis build --config=ui5.yaml --clean-dest --dest dist

info ProjectBuilder Preparing build for project project1
info ProjectBuilder Target directory: dist
info ProjectBuilder Cleaning target directory...
info Project 1 of 1: > Building application project project1...
info project1 > Running task escapeNonAsciiCharacters...
info project1 > Running task replaceCopyright...
info project1 > Running task replaceVersion...
info project1 > Running task minify...
info project1 > Running task generateFlexChangesBundle...
info project1 > Running task generateComponentPreload...
warn lbt:bundle:Builder Module project1/utilis/locate-reuse-libs.js requires top level scope and can only be embedded as a string (requires 'eva
info ProjectBuilder Build succeeded in 793 ms
info ProjectBuilder Executing cleanup tasks...

Confirmation is required to deploy the app:

Application Name: ZIG_NROBJ5
Package: Z_D0538811
Transport Request: X05K900059
Destination: abap-cloud-x05_0000-abapcp-integrationtests-staging-provisioning-tests
SCP: true

✓ Start deployment (Y/n)?
... yes
```

4. After filling in the mandatory fields of the project attributes, add your deployment configuration as well as your FLP configuration by choosing *Yes*.

### ⓘ Note

This configuration can also be added later via the terminal by running the command

```
npx fiori add flp-config
```

Press [Next](#).

5. 1. Add a target repository name and a target package, then create a workbench transport request or choose an already existing one. Press [Next](#).
6. Finally, adjust the Fiori Launchpad Configuration information:
  - Choose a semantic object that represents a business entity such as a customer, a sales order or a product.
  - Choose an action that describes which operation (such as `display` or `manage`) is intended to be performed on a semantic object.
  - Choose a title that will represent the tile name.

Press [Finish](#).

After creating a new SAP Fiori application in BAS, you can deploy it.

### Deploy your SAP Fiori application

1. In the project folder open the created project in the integrated terminal.
2. Go to the `controller` folder which you will find in the folder `webapps` of the project. Open `View1.controller.js`. Insert the code snippet into the view. Adopt the value of the `objectname` variable to the object which you would like to call in Fiori.
3. Deploy your application by means of running the terminal command: `npm run deploy`.
4. After successful deployment a note will appear that is required for the identity application manager.

### ⓘ Sample Code

```
info builder:custom deploy-to-abap Fiori Launchpad App Descr Item  
ZIG_NROBJ5_UI5R was created
```

5. Now, a successfully deployed application will appear in ABAP Development Tool in the BSP library folder of the deployment package.

To publish the business catalog to the system and pin your final product to the home screen of your Fiori Launchpad, continue reading the next topic called [IAM App Creation and Settings](#).

## 4.2.11.6.2.5.1 IAM App Creation and Settings

To publish a business catalog, you need to create a new IAM app first in case you do not use an already existing one.

1. Create a new IAM app under the Cloud Identity and Access Management folder.
2. Add an IAM app name as well as a description and press [Next](#).
3. Choose a corresponding transport request and press [Finish](#).
4. Open the created IAM app and add the previously deployed Business Application Studio app as [Fiori Launchpad Descr Item ID](#). Finally, press [Publish Locally](#).

- Now assign the IAM app to a new or already existing business catalog and an according description. Press **Finish**.
- Complete the process by publishing the business catalog to the system. Press **Publish Locally**.

## Business Role Settings

- Log onto the staging landscape of the Steampunk system and create a new business role or use an existing one.
- Assign the previously published business catalog to the business role and press **Save**.
- In order to run your application properly, your business role should contain the following business catalogs:

Business Catalog ID	Status	Dependencies
SAP_CA_BC_IC_LND_NUM_PIC	Read Only	0
SAP_CA_BC_IC_LND_PIC		0
ZBC_ZHO_NR0105		0

- Go back to the Fiori Launchpad Home view and find your application by searching via App Finder in the drop-down menu of your account icon on the upper right corner of the screen. Now you can also pin the app on your home screen.

## 4.2.11.6.3 Getting Numbers from an Interval

The CL\_NUMBER RANGE \_RUNTIME class provides methods for getting numbers from an interval at runtime.

### Checking Numbers for External Intervals

Use the NUMBER\_CHECK method to check whether a number is within an external interval.

Import Parameters

Parameter Name	Field Name	Value Help
NR_RANGE_NR		Interval number
NUMBER		Number to be checked
NUMERIC_CHECK		Numeric check (for numeric intervals only)
OBJECT		Number range object
SUBOBJECT		Sub-object
TOYEAR		To fiscal year

Parameter Name	Field Name	Value Help
NUMBER_ALPHA		Alphanumeric and optional length check
LENGTH_CHECK		Check of number length
Export Parameter		
Parameter Name	Field Name	Value Help
RETURNCODE		Return code

## Getting Numbers for Internal Intervals

Use the `NUMBER_GET` method to determine the next number of a number range interval.

Import Parameters

Parameter Name	Field Name	Value Help
IGNORE_BUFFER		Ignore Buffer
NR_RANGE_NR		Interval Number
OBJECT		Number Range Object
QUANTITY		Number of Numbers in Buffer
SUBOBJECT		Sub-object
TOYEAR		To Fiscal Year

Export Parameters

Parameter Name	Field Name	Value Help
NUMBER		Returned number
RETURNCODE		Return code
RETURNED_QUANTITY		Number of returned numbers

### ↔ Sample Code

```
...
  lv_object = 'Z_TEST_03'.
...
  CALL METHOD cl_numberrange_runtime=>number_get
    EXPORTING
      nr_range_nr = '01'
      object      = lv_object
    IMPORTING
      number      = DATA(lv_number)
```

```
returncode = DATA(lv_rcode).
```

```
...
```

## Getting the Number Status of an Interval

Use the NUMBER\_STATUS method to determine the number status of a number range interval.

Import Parameters

Parameter Name	Field Name	Value Help
NR_RANGE_NR		Interval number
OBJECT		Number range object
SUBOBJECT		Sub-object
TOYEAR		To Fiscal Year

Export parameter

Parameter Name	Field Name	Value Help
NUMBER		Number status of the interval

### 4.2.11.6.4 Getting Numbers from an Interval in RAP environment

The CL\_NUMBERRANGE\_BUFFER class provides methods for getting numbers from an interval at runtime. Each method is called based on the buffer type setting of the object. In case the method caller calls a method, which doesn't coincide with the buffer type of the object, an exception CX\_NUMBER\_RANGES will be raised.

## Getting Numbers for Internal Intervals

Based on the buffering settings of the object, you can call a specific API method.

### Getting Numbers for the Object with main Memory Buffering

Use the NUMBER\_GET\_MAIN\_MEMORY method to determine the next number of a number range interval.

Import Parameters

Parameter Name	Value Help
IV_OBJECT	Number Range Object

Parameter Name	Value Help
IV_SUBOBJECT	Sub-object
IV_INTERVAL	Interval Number
IV_QUANTITY	Number of Numbers in Buffer
IV_TOYEAR	To Fiscal Year

Export Parameters

Parameter Name	Value Help
EV_NUMBER	Returned Number
EV_RETURNCODE	Return Code
EV_RETURNED_QUANTITY	Number of Returned Numbers

#### ↳ Sample Code

```

lv_object = 'ZIG_NR5'.
lv_interval = '01'.
lv_quantity = 10.
TRY.
DATA(lo_get_number) = cl_numberrange_buffer->get_instance( ).
lo_get_number->if_numberrange_buffer~number_get_main_memory( EXPORTING
iv_object      =
lv_object
iv_subobject = lv_subobject
                           iv_interval = lv_interval
                           iv_quantity =
lv_quantity
                           iv_toyear    = lv_year
                           IMPORTING
ev_number      = lv_number
                           ev_returned_quantity =
lv_returned_qunatity).
CATCH cx_number_ranges INTO DATA(lr_error).
ENDTRY.
```

## Getting Numbers for the object with parallel buffering

Use NUMBER\_GET\_PARALLEL method to determine the next number of a number range interval.

Import Parameters

Parameter Name	Value Help
IV_OBJECT	Number Range Object
IV_SUBOBJECT	Sub-object
IV_INTERVAL	Interval Number
IV_QUANTITY	Number of Numbers in Buffer
IV_TOYEAR	To Fiscal Year

#### Export Parameters

Parameter Name	Value Help
EV_NUMBER	Returned Number
EV_RETURNCODE	Return Code

#### ↔ Sample Code

```
lv_object = 'ZIG_NR2'.
lv_interval = '01'.
TRY.
DATA(lo_get_number) = cl numberrange_buffer->get_instance( ).
lo_get_number->if_numberrange_buffer~number_get_parallel( EXPORTING
iv_object      = lv_object
                           iv_interval   =
lv_interval
                           iv_subobject  =
lv_subobject
                           iv_toyear     =
lv_year
                           IMPORTING
ev_number      = iv_number ).
CATCH cx_number_ranges INTO DATA(lr_error).
ENDTRY.
```

### Getting Numbers for the Object without Buffering

Use the NUMBER\_GET\_NO\_BUFFER method to determine the next number of a number range interval.

#### Export Parameters

Parameter Name	Value Help
EV_NUMBER	Returned Number
EV_RETURNCODE	Return Code
EV_RETURNED_QUANTITY	Number of Returned Numbers

#### ↔ Sample Code

```
lv_object = 'ZIG_NR9'.
lv_interval = '01'.
lv_quantity = 10.
TRY.
DATA(lo_get_number) = cl numberrange_buffer->get_instance( ).
lo_get_number->if_numberrange_buffer~number_get_no_buffer( EXPORTING
iv_object      = lv_object
                           iv_subobject   =
lv_subobject
                           iv_interval    =
lv_interval
                           iv_toyear      =
lv_toyear
                           iv_quantity   =
lv_quantity
                           iv_ignore_buffer =
abap_true
                           IMPORTING
ev_number      = lv_number
                           ev_returned_quantity = lv_returned_qunatity).
CATCH cx_number_ranges INTO DATA(lr_error).
```

```
ENDTRY.
```

## 4.2.11.7 Customer Data Browser

With this app, you can view SAP BTP ABAP environment data owned by the customer.

### Key Features

You can use this app as a self-service to:

- View business data owned by customers.
- View the data of custom fields.
- View the technical names and business names of fields.
- Configure filter criteria and sort data.
- Configure display columns.
- Get the total number of records.
- Export data visible in the application to a spreadsheet.
- Provide controlled and restricted access to users (Admin user feature).

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-CDB.

### Supported Device Types

- Desktop
- Tablet

### Related Information

[Application Overview \[page 1491\]](#)

[Read Access Logging \[page 1492\]](#)

[Security Audit Log \[page 1492\]](#)

[Getting Started \[page 1493\]](#)

[Restrictions and App Behavior \[page 1494\]](#)

[Using the App \[page 1498\]](#)

[FAQs \[page 1499\]](#)

## 4.2.11.7.1 Application Overview

Customer Data Browser is a self-service application that you can use to view data that belongs to CDS views and database tables allowed for user access by SAP or by the customer.

The CDS views and database tables allowed for user access by SAP in the Customer Data Browser app are called Customer Data Browser objects.

You can create Customer Data Browser objects using the ABAP Development Tools (ADT). For more information, see [Working with Customer Data Browser Objects](#).

## CDS View Coverage

In keeping with the SAP BTP ABAP Environment guidelines, the Customer Data Browser application primarily displays data using published CDS views (of the type contract C1) that are relevant to this environment.

For application areas which do not have good coverage of published CDS views, SAP may provide database tables as a temporary solution to display the data. After the relevant CDS views are published, SAP may disable the corresponding tables.

As part of continuous review and update of application content, the number of allowed Customer Data Browser objects available in the application are projected to increase with every upcoming release.

### Disclaimer

The use of the Customer Data Browser application may conflict with the data protection policies of your organization. The authorization concept does not support any filtering by the Data Controller or by typical attributes reflecting the purposes of processing. In addition, a download of the data is a transfer in a software/environment that is potentially not secured. Hence, before using the Customer Data Browser application, please ensure that you follow the data protection Policies in your organization.

## Related Information

[Customer Data Browser \[page 1490\]](#)

[Read Access Logging \[page 1492\]](#)

## 4.2.11.7.2 Read Access Logging

### Use

In Read Access Logging (RAL), you can configure which read-access information to log and under which conditions.

SAP delivers sample configurations for applications.

The **BC/BC-SRV-APS-CDB/Template for Preview Access** captures information related to viewing and accessing data.

In the following configurations, fields are logged in combination with additional fields, in the following business contexts:

Channel	Configuration	Fields Logged	Business Context
SAP Gateway (OData)	BC/BC-SRV-APS-CDB/ Template for Preview Access	Username (SY-UNAME) Request URL (:Request URL)>	Log viewing and accessing data along with the search fil- ter criteria

#### ⓘ Note

See SAP Note [3351781](#) for information about the template RAL configuration file for Customer Data Browser.

### Related Information

[Customer Data Browser \[page 1490\]](#)

[Security Audit Log \[page 1492\]](#)

## 4.2.11.7.3 Security Audit Log

In Security Audit Log (SAL), you can get information about user access to the *Customer Data Browser* application.

You can check the security audit logs for the Customer Data Brower app using the [Display Security Audit Log \[page 2775\]](#) app. In the *Display Security Audit Log* app, choose the audit event FU5.

For more information about security audit logs, see: [Security Audit Log](#).

## Related Information

[Customer Data Browser \[page 1490\]](#)

[Getting Started \[page 1493\]](#)

### 4.2.11.7.4 Getting Started

Customer Data Browser is a self-service application that you can use to view data that belongs to specific CDS views and database tables that are allowed for user access by SAP or by the customer.

#### ⓘ Note

The CDS views and database tables that are allowed for user access by SAP or by the customer in the Customer Data Browser app are called Customer Data Browser objects.

## Viewing Fiori Tiles

This application is available on the SAP Fiori launchpad screen as a customer self-service.

Based on your assigned business role catalog, you can view the SAP Fiori launchpad tile *Customer Data Browser*.

## Required Users

The required user and the respective business catalog role and business role template for the **Customer Data Browser** app are described below.

Required User	User Task	Business Catalog Role	Business Role Template	Application
Business User	Access application	SAP_CORE_BC_CDB_PC	SAP_BR_ADMINISTRATOR_DANA (Administrator - Data Analysis)	Customer Data Browser
Admin User	Maintain authorization restrictions	Not relevant	SAP_BR_ADMINISTRATOR (Administrator)	Maintain Business Roles

## Role Purpose

The roles are used for the following purposes:

- SAP\_BR\_ADMINISTRATOR (Administrator): With this role, you can maintain authorization restrictions using the [https://help.sap.com/docs/SAP\\_S4HANA\\_CLOUD/a630d57fc5004c6383e7a81efee7a8bb/365b0d6fe67d4cecbfe11cc8344264cc.html](https://help.sap.com/docs/SAP_S4HANA_CLOUD/a630d57fc5004c6383e7a81efee7a8bb/365b0d6fe67d4cecbfe11cc8344264cc.html) app.  
For more information about maintaining restrictions for Customer Data Browser, see [How to Define Authorizations Based on Restrictions \[page 2672\]](#).  
For more information about restriction fields for Customer Data Browser, see [Restrictions and App Behavior \[page 1494\]](#).
- SAP\_BR\_ADMINISTRATOR\_DANA (Administrator – Data Analysis): With this role, you can access Customer Data Browser application.

## Role Assignment Recommendations

- Assign the SAP\_BR\_ADMINISTRATOR\_DANA role to a business user who does not have the SAP\_BR\_ADMINISTRATOR role.
- Make sure that the user with the SAP\_BR\_ADMINISTRATOR\_DANA role is assigned to a single copy of the SAP\_BR\_ADMINISTRATOR\_DANA role. Assigning the user to multiple copies of the SAP\_BR\_ADMINISTRATOR\_DANA role may cause authorization issues when you access the Customer Data Browser app.

## Related Information

[Customer Data Browser \[page 1490\]](#)

[Restrictions and App Behavior \[page 1494\]](#)

## 4.2.11.7.5 Restrictions and App Behavior

### Restrictions Required to Be Maintained by the Administrative User

The *Read/Value Help* restriction type for the Customer Data Browser app consists of the following restriction fields:

Restriction Field	Description
Object Group	<p>Logically groups Customer Data Browser objects.</p> <p>Note that this field is not active.</p>
Object Name	Displays a list of Customer Data Browser objects and their descriptions.
Object Fields	<p>Provides access to the following categories of fields in Customer Data Browser objects:</p> <ul style="list-style-type: none"> <li>• GE: General fields that do not contain sensitive personal information</li> <li>• SP: Sensitive personal information fields that contain key fields and fields with sensitive personal information</li> <li>• Unrestricted: All fields containing general and sensitive personal information</li> </ul> <p>Note that the field categories are determined by internal SAP classifications.</p>
Object Records	<p>Provides access to the following categories of data records in Customer Data Browser objects:</p> <ul style="list-style-type: none"> <li>• OV: Records with bypassed access controls (privileged access)</li> <li>• AC: Records that fulfill access controls (unprivileged access)</li> <li>• Unrestricted: Records that fulfill access controls (unprivileged access)</li> </ul>

### ① Note

When maintaining restrictions, set the [Write, Read, Value Help](#) restriction to **No Access**.

## App Behavior in Response to Administrative Restrictions

Refer to the restriction matrix below to understand the expected behavior of the application.

Restrictions Set By the Administrator			Display Behavior in Customer Data Browser App				
For Object Name	For Object Field	For Object Record	Objects	CDS Views	Database Tables	Fields	Records

Restrictions Set By the Administrator			Display Behavior in Customer Data Browser App				
Unrestricted access	Unrestricted	Unrestricted	Displays all objects	Displays data that is authorized for the user. Note that the user is explicitly required to obtain the necessary authorizations.	No data displayed	Displays general (GE) and sensitive personal (SP) fields	Displays records that fulfill access controls (AC)
Restrictions not maintained	Restrictions not maintained	Restrictions not maintained	Displays no objects	No data displayed	No data displayed	No data displayed	No data displayed
Restricted access	Access to general fields (GE)	Access to records that fulfill access controls (AC)	Displays objects with access	Displays data that is authorized for the user. Note that the user is explicitly required to obtain the necessary authorizations.	No data displayed	Displays general fields (GE)	Displays records that fulfill access controls (AC)
Restricted access	Access to sensitive personal fields (SP)	Access to records that fulfill access controls (AC)	Displays objects with access	Displays data that is authorized for the user. Note that the user is explicitly required to obtain the necessary authorizations.	No data displayed	Displays sensitive personal fields (SP)	Displays records that fulfill access controls (AC)
	Access to general (GE) and sensitive personal (SP) fields	Access to records that fulfill access controls (AC)	Displays objects with access	Displays data that is authorized for the user. Note that the user is explicitly required to obtain the necessary authorizations.	No data displayed	Displays general (GE) and sensitive personal (SP) fields	Displays records that fulfill access controls (AC)

Restrictions Set By the Administrator		Display Behavior in Customer Data Browser App						
		Access to general fields (GE)	Access to records with bypassed access controls (OV)	Displays objects with access	Displays all data	Displays all data	Displays general fields (GE)	Displays all records
Restricted Read Access	Access to sensitive personal fields (SP)	Access to records with bypassed access controls (OV)	Access to records with bypassed access controls (OV)	Displays objects with access	Displays all data	Displays all data	Displays sensitive personal fields (SP)	Displays all records
Restricted access	Access to general (GE) and sensitive personal (SP) fields	Access to records with bypassed access controls (OV)	Access to records with bypassed access controls (OV)	Displays objects with access	Displays all data	Displays all data	Displays general (GE) and sensitive personal (SP) fields	Displays all records

## Summary of User Access

### Business Users

You can access...	If...	Even if...
Data in CDS views	You have the required CDS view-specific access controls	You have Customer Data Browser-specific access to bypass access controls (Object Records: OV)
Data in database tables	You have Customer Data Browser-specific access to bypass access controls (Object Records: OV)	

### Administrators

You can restrict access to Customer Data Browser objects by providing a list of object names in the *Object Name* restriction field.:

## Related Information

[Customer Data Browser \[page 1490\]](#)

[Using the App \[page 1498\]](#)

## 4.2.11.7.6 Using the App

When you choose the **Customer Data Browser** tile, you can see a list of Customer Data Browser objects that you are authorized to view.

The list of Customer Data Browser objects includes:

- CDS views, including custom CDS views
- Database tables

### Working with the App

#### Navigating to the Data Details Screen

Choose a row to navigate to the *Customer Data Browser – Data Details* screen where you can view the data relevant to the object. You can sort the data and configure filter criteria to view a subset of data.

#### Entering Parameter Values for CDS Views

If you choose a CDS view that requires you to enter parameter values before previewing data, you can enter the values in the Maintain Parameters dialog box.

#### Customer Data Browser - Data Details Screen

You can see the following options on the *Customer Data Browser – Data Details* screen

- **Total Records:** Choose to display the total number of records in the selected Customer Data Browser object.

##### ⓘ Note

If you set filters, the total number of records is calculated based on the filter criteria.

- **View Technical Name/View Business Name:** Choose this toggle button to display the technical names or the business names of the columns.
- **Parameters:** Choose this button to change parameter values for CDS views that require parameters.

##### ⓘ Note

This button is visible if the CDS view requires parameter values before preview.

- **Maximum Rows to Be Displayed:** Enter the maximum number of rows to be displayed on the screen. By default, this value is set to 100. This means that the first 100 entries that match the filter criteria are displayed on the screen. You can reset this value to display a different maximum number.
- **Adapt Filters:** Choose this button to select additional filters or deselect existing filters.
- **Export :** From the *Export* icon, choose the export option to download data displayed on the screen to your local machine as a spreadsheet.
- **Settings:** Choose this button to select columns you want to view or deselect columns you no longer want to view.

## Related Information

[Customer Data Browser \[page 1490\]](#)

[FAQs \[page 1499\]](#)

## 4.2.11.7.7 FAQs

### Business Users

#### 1. Why can't I see any Customer Data Browser objects in the app?

**Answer:** You do not have the authorization to view the objects. Please contact your administrator and ask them to grant the authorization you need to view the objects. For more information, see [Restrictions and App Behavior \[page 1494\]](#).

#### 2: How can I see the complete list of Customer Data Browser objects available with the app?

**Answer:** You can view the complete list of Customer Data Browser objects if you have admin-level authorization. Contact your administrator to get the complete set of authorizations. Please ask the administrator to select all values in the *Object Name* restriction field.

#### 3. Why can't I find a particular CDS view or table in the list of Customer Data Browser objects?

**Answer:** Here are some possible reasons:

- Your administrator has not granted you authorization to access this object, or
- The particular object is not yet enabled for Customer Data Browser

Contact your administrator for the exact reason.

#### 4. When trying to view the CDS view data, why do I get the message "You either do not have the required authorization or there is no data to be displayed"?

**Answer:** You get this message because:

- Your administrator has not granted the required CDS view-specific authorizations (business catalogs) to your user. This means you cannot view the data of this CDS view.
- Alternatively, your administrator has granted the required CDS view-specific authorizations (business catalogs) to your user, but there is no data to be displayed. So, the system displays the error message.

Contact your administrator to obtain the required CDS view-specific authorizations.

#### 5. When trying to view the data of a table, why do I get the message "No authorization to view object data"?

**Answer:** You see this message when your administrator has not granted the Customer Data Browser-specific authorizations to your user to view the data of database tables.

Contact your administrator to obtain the required authorizations.

For more information, see [Restrictions and App Behavior \[page 1494\]](#).

## **6. My administrator has maintained the required restrictions to view data of relevant tables or CDS views. However, I keep getting a message in the app that I am not authorized to view the data.**

**Answer:** Inform your administrator that you do not have the authorization to view the specific tables or CDS views.

- The user with the SAP\_BR\_ADMINISTRATOR\_DANA role is assigned to a single copy of the SAP\_BR\_ADMINISTRATOR\_DANA role (either the original or a copy of original, we recommend creating a copy). You can check the assignment in the [Maintain Business Users \[page 2655\]](#) app.
- The *Write, Read, Value Help* restriction is maintained as **No Access**

To validate the values provided in the restrictions, see [Restrictions and App Behavior \[page 1494\]](#).

## **Administrators**

### **1. As an administrator, how can I maintain restrictions for the Customer Data Browser app?**

**Answer:** For detailed information about restrictions, see [Restrictions and App Behavior \[page 1494\]](#).

### **2. As an administrator, how can I identify the CDS view-specific authorizations so that the user can view the data of a specific CDS view?**

**Answer:** In the [SAP Business Accelerator Hub](#), choose *CDS Views* to see detailed information about published interface (I\_) CDS views.

If you have any issues identifying the necessary CDS view-specific authorizations, report an incident under the application component relevant to the CDS view.

### **3 I have maintained the required restrictions for the business user to view data of relevant tables or CDS views. However, the business user keeps getting a message in the app that there is no authorization to view the data.**

**Answer**

- Ensure that the user with the SAP\_BR\_ADMINISTRATOR\_DANA role is assigned to a single copy of the SAP\_BR\_ADMINISTRATOR\_DANA role (either the original or a copy of original, we recommend creating a copy). You can check the assignment in the [Maintain Business Users \[page 2655\]](#) app.
- Ensure that the *Write, Read, Value Help* restriction is maintained as **No Access**

To validate the values provided in the restrictions, see [Restrictions and App Behavior \[page 1494\]](#).

### **4: How can I grant authorization to the business user to view the complete set of Customer Data Browser objects available with the app?**

**Answer:** You can provide unrestricted access to the business user by choosing the *Unrestricted Access* buttons in the *Object Name* window.

## Related Information

[Customer Data Browser \[page 1490\]](#)  
[Application Overview \[page 1491\]](#)  
[Read Access Logging \[page 1492\]](#)  
[Security Audit Log \[page 1492\]](#)  
[Getting Started \[page 1493\]](#)  
[Restrictions and App Behavior \[page 1494\]](#)  
[Using the App \[page 1498\]](#)

### 4.2.11.8 Application Jobs

You can use this app to schedule and monitor application-related jobs. If you have manual activities that you often need to do at a specific time, the [Application Jobs](#) app can reduce your workload by running these tasks smoothly in the background. Planning regular jobs will keep you free to concentrate on other tasks.

The application job framework ideally defines long-running, repeated jobs. It's not recommended to use it to define short-running jobs with a high frequency. See also [Technical Limits and Boundary Conditions for ABAP Applications](#).

## Key Features

You can use this app to:

- Schedule jobs based on a job template: Select a job template, define the description, start date, start time and recurrence of the job, and enter job-specific selection criteria and parameters
- Work with personalized job templates: Create and personalize a job template, save personalized job template for later use, and share personalized template

#### ⓘ Note

The parameters under [Scheduling Information](#) can't be saved as part of a template.

- Schedule jobs with a custom factory calendar: In the [Application Jobs](#) app, define the scheduling information, change an existing calendar, create a new one, and define execution days
- Arrange job series that take different time zones into account
- Monitor jobs
- Display logs and results: You can navigate to job logs and job results by selecting the icon in the [Log](#) or the [Results](#) column
- Copy a job
- Cancel a job
- Restart job chains
- Navigate to the [Application Logs](#) reuse library. You can do this by selecting the corresponding icon in the log on the initial screen of the app.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-APJ.

## Related Information

[Application Logs \[page 1536\]](#)

[Working with Application Job Objects](#)

### 4.2.11.8.1 Creating a Job Catalog Entry and a Job Template

Follow these steps to run your business coding in an application job.

#### Procedure

1. Implement the business logic. You need to create a class which implements certain interfaces so that it's usable in an application job. Implement the coding there that should run within the application job. For more information, see [Implementing the Business Logic \[page 1503\]](#).
2. Define the job catalog entry. Using ABAP development tools for Eclipse, you create a job catalog entry which refers to the class of step 1. In the job catalog entry, you can specify how the selection fields are rendered in the job scheduling dialog in the app. In addition, you can configure various settings which are used by the application job. For more information, see [Creating the Job Catalog Entry \[page 1508\]](#).
3. Define the job template. Using ABAP development tools for Eclipse, you create a job template which refers to the job catalog entry of step 2. In the job template, you define the default values of the job parameters which appear on the selection screen. For more information, see [Creating the Job Template \[page 1509\]](#).

For more information on how to create a job catalog entry and a job template in ABAP development tools for Eclipse, see [Working with Application Job Objects](#).

If you want to implement a simple RAP (ABAP RESTful application programming model) business object, see [Code Sample: How to Schedule Application Jobs from a RAP-based Business Object](#). This code sample allows you to schedule a class as an application job that takes the semantic key of the selected entity as a parameter.

## Related Information

[Setting up the Authorizations \[page 1525\]](#)

### 4.2.11.8.1.1 Implementing the Business Logic

Business logic is ABAP coding that you want to be run within an application job.

Usually, business logic also defines the data type and some properties of the selection fields that you get on the selection screen before you schedule an application job. In this context, the business logic is implemented in an ABAP class. The following development steps require a user with the development role.

To implement the business logic, create an ABAP class that implements a certain interface:

- Create an ABAP class with interface `IF_APJ_RT_RUN`. Here, the selection fields are defined as public attributes of the class. See [Creating an ABAP Class with Interface IF\\_APJ\\_RT\\_RUN \[page 1504\]](#) for more information. It's recommended to use this interface for all new developments.
- You can still create an ABAP class with interfaces `IF_APJ_DT_EXEC_OBJECT` and `IF_APJ_RT_EXEC_OBJECT`. Here, the selection fields are defined in method `GET_PARAMETERS` of the interface `IF_APJ_DT_EXEC_OBJECT`. See [Creating an ABAP Class with Interfaces IF\\_APJ\\_DT\\_EXEC\\_OBJECT and IF\\_APJ\\_RT\\_EXEC\\_OBJECT \[page 1506\]](#) for more information.

#### Note

Note that this is a legacy interface. Don't use it for new developments. If you use this interface, you only have limited options during the editing of the job catalog entry.

The ABAP class mentioned here is considered the main class of each application job. This class needs to be part of a customer-defined software component (not `ZLOCAL`) to be able to transport the class into subsequent systems.

## Related Information

[Creating the Job Catalog Entry \[page 1508\]](#)

[Creating the Job Template \[page 1509\]](#)

[Setting up the Authorizations \[page 1525\]](#)

[Application Logs \[page 1536\]](#)

## 4.2.11.8.1.1.1 Creating an ABAP Class with Interface IF\_APJ\_RT\_RUN

Learn how to create an ABAP class that implements the interface `IF_APJ_RT_RUN` to run the business logic of your application in an application job.

The interface `IF_APJ_RT_RUN` contains the following method:

- `EXECUTE`: This method is called by the application job framework when the scheduled application job is run. This means that the ABAP coding that should run within the application job is implemented here.

Before you schedule an application job, you get a selection screen where you can enter values for the application job parameters. These parameters are used to control the processing of the business logic. If you put the business logic into an ABAP class with interface `IF_APJ_RT_RUN`, you can use the public attributes of the ABAP class as selection parameters. The parameters should fulfil the requirements described in [Defining Class Attributes Used as Selection Parameters \[page 1504\]](#). Apart from the job catalog entry, the data type of the class attribute determines how the parameter is displayed and behaves on the selection screen.

Before the `EXECUTE` method is called by the application job framework during the job run, the class attributes are automatically filled with the values which were entered on the selection screen. The attributes can be used in the class coding without further preparations.

When you create a new application job template based on this class (see [Creating the Job Template \[page 1509\]](#)), its parameter values are prefilled with default values. To determine these default values, the class is instantiated and the values of the class attributes which have the same name as the template parameters are read. You have the following possibilities to set these default values:

- In the definition section of the class as default values of the class attributes
- In the `CONSTRUCTOR` of the class by setting values to the class attributes
- In addition, you can implement the interface `IF_APJ_DT_DEFAULTS` in the class and define default values in method `FILL_ATTRIBUTE_DEFAULTS` by setting values to the class attributes

### ⓘ Note

If the parameter is a date field, you can use a command like `@STARTDAYPLUSDAY@ : 03` in the value field of the template to define a dynamic date. This command can't be set as default value in method `FILL_ATTRIBUTE_DEFAULTS`, but you can enter it into the template after the template has been created.

## 4.2.11.8.1.1.1.1 Defining Class Attributes Used as Selection Parameters

Find out how to define class attributes that can be used as selection parameters to create your own business logic.

Before you schedule an application job, you get a selection screen where you can enter the values of the selection parameters. These values are transferred to the ABAP class before the `EXECUTE` method is run by the application job. To use a class attribute as selection parameter of the application job, it should fulfil the following requirements:

- The class attribute must be public
- Constants and read-only attributes can't be used
- The length of the attribute name is limited to 38 characters

#### ⓘ Note

If the attribute is part of an interface which is implemented in the class, and if the total length of the interface name and attribute name is longer than 38 characters, you may define an alias in the class and use the alias name in the job catalog entry

On the selection screen, fields either allow a single value, or multiple values or a value range to be entered. These fields have the following additional requirements:

- If the selection field allows the input of a single value, the corresponding class attribute must have an elementary data type.

#### ⓘ Note

An elementary data type is a data type that contains a single value. Examples are built-in data types like C or I, or data elements which are defined in the data dictionary.

The data types STRING and XSTRING are allowed. Mind that the application job framework only allows values with a maximum length of 255 characters. If a character field with a length such as 1024, or a string is used, and if you enter a value which is longer than 255 characters on the selection screen, the value is cut off.

References and complex data types like structures or internal tables are not allowed because they can't be represented by a selection field with a single value.

- If the selection field allows the input of multiple values or a value range, the class attribute must be defined as ranges table.

#### ⓘ Note

A ranges table is an internal table with the fields SIGN, OPTION, LOW, and HIGH. These fields fulfil the following conditions:

- SIGN is a character field of length 1
- OPTION is a character field of length 2
- LOW uses an elementary data type
- HIGH uses the same data type as LOW

A ranges table can, for example, be defined via ABAP command TYPE RANGES OF.

The data type of the class attribute determines the data type and behavior of the selection parameter on the selection screen. An example: if the class attribute uses a date type, the selection field allows to enter a date, and it offers a value help for date selection.

The data type of the class attribute determines whether the selection field on the selection screen allows to enter a single value or multiple values or value ranges. If the data type is an elementary type, a single value can be entered. If it's a ranges table, multiple values or value ranges can be used.

When a parameter is added to a job catalog entry, the default value of the parameter title is taken from the description text of the class attribute. In ABAP development tools for Eclipse, the description text of the

class attribute can be set using the ABAP Doc comment: " ! <p class="shorttext synchronized" lang="en">...</p>

It's possible to define class attributes in the ABAP class which don't fulfil the above requirements if you don't want to use them as parameters of a job catalog entry.

## 4.2.11.8.1.1.2 Creating an ABAP Class with Interfaces IF\_APJ\_DT\_EXEC\_OBJECT and IF\_APJ\_RT\_EXEC\_OBJECT

Find out how to run business logic in an application job using an ABAP class that implements interfaces `IF_APJ_DT_EXEC_OBJECT` and `IF_APJ_RT_EXEC_OBJECT`.

### ⓘ Note

Note that this is a legacy interface. Don't use it for new developments. If you use this interface, you only have limited options during the editing of the job catalog entry.

If you want to run the business coding of your application in an application job, you can create an ABAP class which implements the interfaces `IF_APJ_DT_EXEC_OBJECT` and `IF_APJ_RT_EXEC_OBJECT` and put the coding into this class. It's not recommended to use this possibility for new developments, because it offers limited functionality only.

The interface `IF_APJ_DT_EXEC_OBJECT` contains the following method:

- `GET_PARAMETERS`: This method is called by the application job framework to get all selection parameters. The method returns two internal tables:
  - `ET_PARAMETER_DEF`: Determines the parameter list which is used by the job catalog entry that refers to this main class. It also defines how the parameters are displayed and how they behave on the selection screen
  - `ET_PARAMETER_VAL`: Determines the default values for these parameters in a newly created job template that refers to the job catalog entry mentioned above

The structure of table `ET_PARAMETER_DEF` has the following fields:

Field Name	Definition
SELNAME	Name of the parameter. The length of the name is restricted to 8 characters. It defines the parameter name which is used in the job catalog entry and in the job template
KIND	Defines whether a single value or a value range can be entered on the selection screen for this parameter. Possible values are: <ul style="list-style-type: none"><li>• <code>IF_APJ_DT_EXEC_OBJECT=&gt;PARAMETER</code> for a single value parameter</li><li>• <code>IF_APJ_DT_EXEC_OBJECT=&gt;SELECT_OPTION</code> for a value range parameter</li></ul>
DATATYPE	The data type of the parameter. It must be an elementary data type, like C or CHAR
LENGTH	The output length of the parameter if the data type has no fixed length
DECIMALS	Number of decimals if the data type belongs to a number with decimals

Field Name	Definition
COMPONENT_TYPE	Not supported
SECTION_TEXT	Not used
GROUP_TEXT	Not used
PARAM_TEXT	The title of the parameter which is displayed on the selection screen
LOWERCASE_IND	Defines whether lower case characters are allowed for the parameter
HIDDEN_IND	Defines whether the parameter is hidden on the selection screen
CHANGEABLE_IND	If set to ABAP_FALSE, the read-only flag is set for this parameter if the parameter is added to a catalog entry
MANDATORY_IND	Defines whether the parameter is mandatory on the selection screen
CHECKBOX_IND	If set, the parameter is displayed as checkbox on the selection screen
LIST_IND	If set, the parameter is displayed as list box on the selection screen
RADIO_GROUP_IND	If set, the parameter is displayed as radio button on the selection screen
RADIO_GROUP_ID	Name of the radio button group if the parameter is a radio button

The structure of the table ET\_PARAMETER\_VAL has the following fields:

Field Name	Definition
SELNAME	Name of the parameter. It's the same name that was used in table ET_PARAMETER_DEF
KIND	Defines whether a single value or a value range can be entered on the selection screen for this parameter. It should be the same value that was used in table ET_PARAMETER_DEF
SIGN, OPTION, LOW, HIGH	Entry of a ranges table which contains the parameter value, or a parameter value range

The interface IF\_APJ\_RT\_EXEC\_OBJECT contains the following method:

- EXECUTE: This method is called by the application job framework when the scheduled job is run. It receives the internal table IT\_PARAMETERS as parameter. This internal table contains all parameter values that you entered on the selection screen when you scheduled the job.

The structure of table IT\_PARAMETERS has the following fields:

Field Name	Definition
SELNAME	Name of the parameter. It's the same name that was used in method GET_PARAMETERS
KIND	If you've defined the parameter as a single value parameter in method GET_PARAMETERS, it's set to IF_APJ_DT_EXEC_OBJECT=>PARAMETER. Otherwise, it's set to IF_APJ_DT_EXEC_OBJECT=>SELECT_OPTION

Field Name	Definition
SIGN, OPTION, LOW, HIGH	Entry of a ranges table which contains the parameter value, or a parameter value range

#### ⓘ Note

If you've selected several value ranges for a parameter on the selection screen, the table `IT_PARAMETERS` contains one entry for each value range.

### 4.2.11.8.1.1.3 Exception Handling

Learn how to handle the exception `CX_APJ_RT_CONTENT` of the method `EXECUTE`.

The method `EXECUTE` has the exception `CX_APJ_RT_CONTENT` in its signature. You can raise an exception of this type and specify a message text (see code snippet). The application job framework handles exceptions of type `CX_APJ_RT_CONTENT` and creates a separate log for the application job. The log contains the message text, and information about the source position where the exception was raised. The application job gets the status *Failed*.

#### ↗ Sample Code

```

" the following coding simulates the application logic that causes a system
exception
" (in this case: division by zero)
DATA a TYPE i.
DATA b TYPE i.
DATA c TYPE i.
TRY.
  a = 1.
  b = 0.
  c = a / b.
  CATCH cx_root INTO DATA(exc).
" Precondition for the following statement:
" message class Z_APJ must exist, message 001 must exist in this message
class with one variable part

  RAISE EXCEPTION TYPE cx_apj_rt_content MESSAGE ID 'Z_APJ' NUMBER 001 WITH
exc->get_text( ).
ENDTRY.

```

The application job framework doesn't handle any other exception apart from `CX_APJ_RT_CONTENT`. Any other exception which is not handled within the `EXECUTE` method itself will lead to a runtime error. The log will then contain the name of the runtime error, and further details of the runtime error have to be checked in the *ABAP Runtime Errors* app.

### 4.2.11.8.1.2 Creating the Job Catalog Entry

An application job catalog entry contains the information that is needed for scheduling and running an application job. The application job catalog entry contains the name of the class that implements the business

logic, and the information on how selection fields are rendered in the job scheduling dialog in the app. In addition, the job catalog entry contains some further settings which are used during the scheduling and running of the application job.

To create a job catalog entry, use the ABAP development tools for Eclipse (ADT) editor. For more information, see [Working with Application Job Objects](#). Alternatively, you could still also create job catalog entries and job templates via a released API. For more information, see [Creating a Job Catalog Entry and a Job Template via API \[page 1523\]](#).

Job catalog entries and job templates are created with a package assignment and the objects are assigned to a transport request. You need to make sure that the package and transport request fit together (regarding the transport layer) and that possible naming conventions are obeyed.

## Related Information

[Creating the Job Template \[page 1509\]](#)

[Setting up the Authorizations \[page 1525\]](#)

### 4.2.11.8.1.3 Creating the Job Template

An *Application Job Template* refers to an application job catalog entry and contains values for some or all selection fields. It can be thought of as a variant of an application job catalog. In the simplest case, a job template doesn't contain any values for the selection fields. In this case, the selection fields are prefilled with their initial values (such as 0 if the selection field is of type integer). An application job catalog entry can have more than one application job template. The job template is the entity that can be scheduled in the *Application Jobs* app. In the scheduling dialog, the selection fields appear and are prefilled with the values defined in the template. They can be overwritten by the user. Scheduling an application job template then results in an application job.

The creation of a job template follows the same technical rules as a *Job Catalog Entry* as described in [Creating the Job Catalog Entry \[page 1508\]](#). To create a job template, use the ABAP development tools for Eclipse. For more information, see [Creating Application Job Templates](#). Alternatively, you could still also create application job templates via a released API. For more information, see [Creating a Job Catalog Entry and a Job Template via API \[page 1523\]](#).

## Related Information

[Setting up the Authorizations \[page 1525\]](#)

## 4.2.11.8.1.4 Example Implementation Using Interface IF\_APJ\_RT\_RUN

This example demonstrates the implementation of the concept of application jobs based on a class with interface `IF_APJ_RT_RUN`.

To implement the business logic, the ABAP class will contain the following attributes that will also be visible on the selection screen when the application job is scheduled:

- The attribute `NUMBERS` which may contain value ranges
- The attribute `NAMES` which may contain multiple single values
- The attribute `TEXT` which may contain a single value only
- The boolean attributes `NUMBERS_AVAILABLE` and `NAMES_AVAILABLE`. The attribute `NUMBERS` will only be used if `NUMBERS_AVAILABLE` is set, while the attribute `NAMES` will only be used if `NAMES_AVAILABLE` is set.
- In addition, the class will contain the attribute `ANOTHER_PUBLIC_ATTRIBUTE` which will not be visible on the selection screen and is not filled by the application job framework

To implement these requirements, the ABAP class is created with the following definition part:

### ↔ Sample Code

```
CLASS zcl_apj_demo_class DEFINITION PUBLIC FINAL CREATE PUBLIC .  
PUBLIC SECTION.  
    INTERFACES if_apj_rt_run.  
    INTERFACES if_apj_dt_defaults.  
TYPES:  
    BEGIN OF ty_name_range,  
        sign TYPE c LENGTH 1,  
        option TYPE c LENGTH 2,  
        low TYPE c LENGTH 50,  
        high TYPE c LENGTH 50,  
    END OF ty_name_range.  
TYPES: ty_name_ranges TYPE STANDARD TABLE OF ty_name_range WITH EMPTY KEY.  
    !<p class="shorttext synchronized" lang="en">Numbers available</p>  
DATA numbers_available TYPE abap_bool VALUE abap_true.  
    !<p class="shorttext synchronized" lang="en">Numbers</p>  
DATA numbers TYPE RANGE OF i.  
    !<p class="shorttext synchronized" lang="en">Names available</p>  
DATA names_available TYPE abap_bool VALUE abap_true.  
    !<p class="shorttext synchronized" lang="en">Names</p>  
DATA names TYPE ty_name_ranges.  
    !<p class="shorttext synchronized" lang="en">Some text</p>  
DATA text TYPE c LENGTH 255.  
    !<p class="shorttext synchronized" lang="en">Public attribute which is  
not used on the selection screen</p>  
    DATA another_public_attribute TYPE c LENGTH 10.  
ENDCLASS.
```

The class definition contains the following parts:

- The class implements the interface `IF_APJ_RT_RUN`. This is necessary so the class can be used in an application job. In addition, it implements the interface `IF_APJ_DT_DEFAULTS` to set default values of new templates, which is optional
- It contains the following public attributes:

- NUMBERS and NAMES are defined as ranges tables, because they may contain more than one value on the selection screen. The example implementation shows two possible ways how the ranges tables can be defined in the class
- Because field TEXT should only contain a single value on the selection screen, it's defined with an elementary datatype in the class
- The attributes NUMBERS\_AVAILABLE and NAMES\_AVAILABLE are defined as boolean type
- Finally, there is attribute ANOTHER\_PUBLIC\_ATTRIBUTE which may have any data type because it will not be visible on the selection screen
- All attributes have description texts which are defined via ABAP Doc comments

In the implementation part of the class, the method IF\_APJ\_RT\_RUN~EXECUTE must be implemented. This method is called by the application job framework when the application job is started. The method should contain the business logic that should be run by the application job.

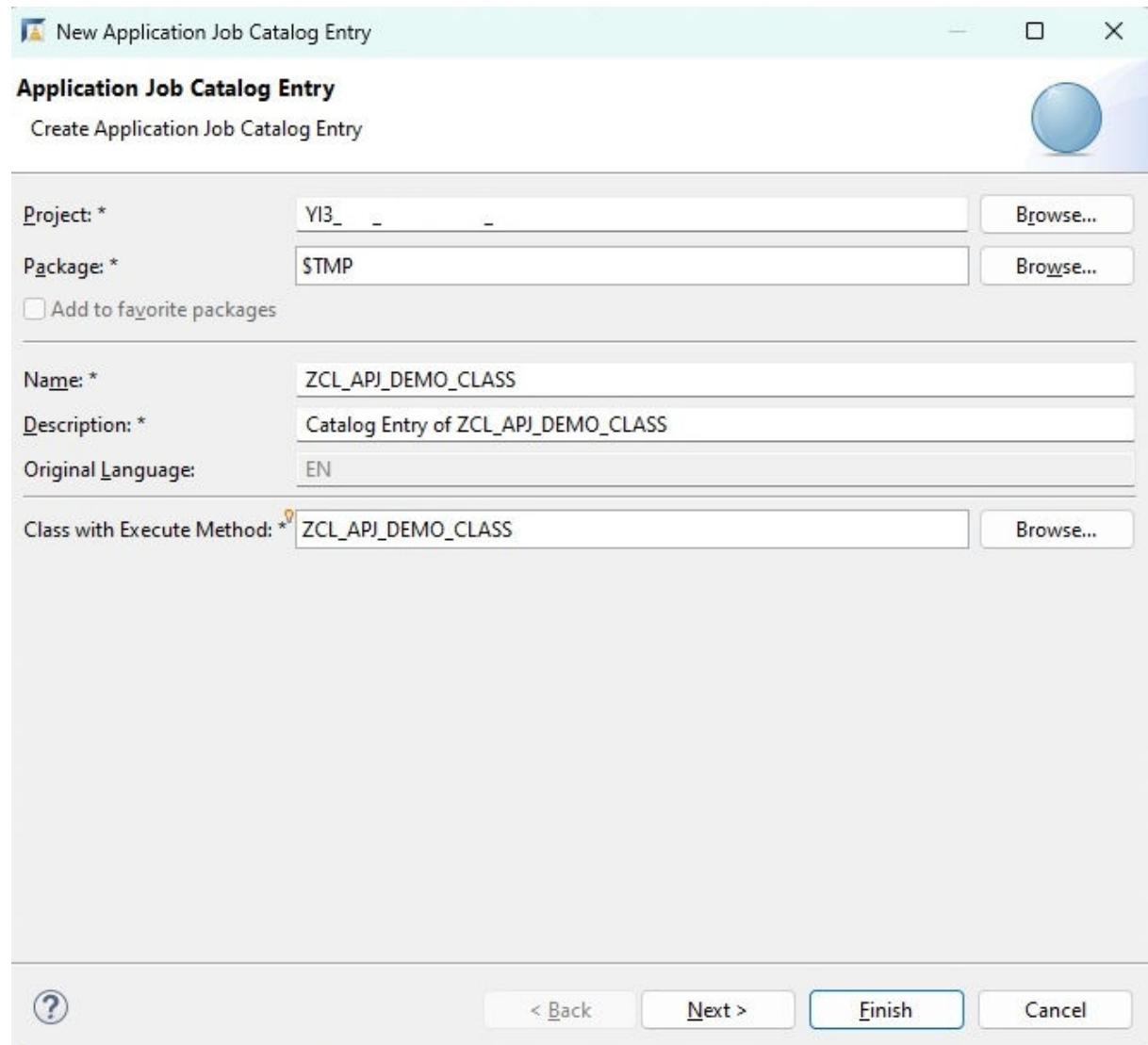
### ↳ Sample Code

```
METHOD if_apj_rt_run~execute.
  TRY.
    DATA(l_log) = cl_bali_log=>create_with_header(
      header = cl_bali_header_setter=>create( object =
'ZOBJECT'
                                         subobject =
'ZSUBOBJECT' ) .
      IF numbers_available = abap_true AND '42' IN numbers.
        l_log->add_item( item = cl_bali_free_text_setter=>create( severity
= if_bali_constants=>c_severity_information
                                         text =
'42 is in the number ranges' ) ).
      ENDIF.
      IF names_available = abap_true AND names IS NOT INITIAL.
        l_log->add_item( item = cl_bali_free_text_setter=>create( severity
= if_bali_constants=>c_severity_status
                                         text =
'Some names are available' ) .
      ENDIF.
      l_log->add_item( item = cl_bali_free_text_setter=>create( severity =
if_bali_constants=>c_severity_status
                                         text = CONV
#( text ) ) .
      cl_bali_log_db=>get_instance( )->save_log_2nd_db_connection( log =
l_log
assign_to_current_appl_job = abap_true ) .
    CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
      " some error handling
    ENDTRY.
  ENDMETHOD.
```

In this example, the EXECUTE method creates an application log and assigns it to the current application job when the job is run. If the number 42 is in the ranges of field NUMBERS, a message is written into the log. Another message is written if field NAMES contains any entry. Finally, the content of field TEXT is put into the log. Attributes like NUMBERS, NAMES and TEXT can be used without any preparation, because they are automatically filled by the application job framework before the EXECUTE method is started.

After the class was implemented, a new catalog entry should be created for this class so that it can be used in an application job. Besides other settings, the catalog entry defines which parameters will be visible on the selection screen (which is visible when you schedule the job) and where and how these parameters are displayed.

A new catalog entry can be created in the ABAP development tools for Eclipse. It can be found under *Other ABAP Repository Object* → *Application Jobs* → *Application Job Catalog Entry*.



On the first screen, you select the name and a description text of the new application job catalog entry and the name of the class which is run within the job. It's the class which you just defined.

After you have created the application job catalog entry, you can start editing it:

Name	Title	Group	Inden...	Mand...	Hidden	Read ...	Enabled By...	Screen Ele...	Radio Butt...	Value Help	Value Help ...	Back...	Only...	Text Lines i...
ANOTHER_PUBLIC_ATTRIBUTE	Public attribute whi...		No	No	No	No		None		None	No	No	0	
NAMES	Names		No	No	No	No		None		None	No	No	0	
NAMES_AVAILABLE	Names available		No	No	No	No		None		None	No	No	0	
NUMBERS	Numbers		No	No	No	No		None		None	No	No	0	
NUMBERS_AVAILABLE	Numbers available		No	No	No	No		None		None	No	No	0	
TEXT	Some text		No	No	No	No		None		None	No	No	0	

In this demo implementation, the table with the parameters was automatically prefilled with all public attributes of the ABAP class which fulfil the requirements to be used in a catalog entry. The titles of the parameters were automatically filled with the description texts of the class attributes (which were defined via the ABAP Doc comments in the definition part of the class).

As a next step, you can change the properties of each parameter and define what the selection screen should look like. In this example, we will meet the following requirements:

- The parameters NUMBERS\_AVAILABLE and NUMBERS will be grouped. The group will get a header text. NUMBERS\_AVAILABLE will be a checkbox, and it will only be possible to enter values into NUMBERS if NUMBERS\_AVAILABLE is set.
- The parameters NAMES\_AVAILABLE and NAMES will be grouped. The group will get a header text. NAMES\_AVAILABLE will be a checkbox, and it will only be possible to enter values into NAMES if NAMES\_AVAILABLE is set.
- Both groups will be displayed side by side in the same section of the screen (NUMBERS on the left side and NAMES on the right side). This section will get a header text, too.
- The parameter TEXT will get its own group (with group header) which is part of its own section (with section header).
- The parameter ANOTHER\_PUBLIC\_ATTRIBUTE is not relevant here and will not be displayed.

To implement these settings, you first define the sections of the selection screen:

Name	Title
SECTION_1	Section with Checkboxes
SECTION_2	Section with Text Field

There are two sections carrying the section header texts *Section with Checkboxes* and *Section with Text Field*. Because section SECTION\_1 is above section SECTION\_2 in this table, section SECTION\_1 is displayed above section SECTION\_2 on the selection screen.

Next, you define the groups:

Groups		Title	Section	
Name	GROUP_1	Group with Numbers	SECTION_1	Add...
GROUP_2	GROUP_2	Group with Names	SECTION_1	Edit...
GROUP_3	GROUP_3	Group with Text Field	SECTION_2	Remove
				Up
				Down

Three groups were defined: The groups with title texts *Group with Numbers* and *Group with Names* were assigned to the first section, while the group with title text *Group with Text Field* was assigned to the second section. Because group GROUP\_1 is above group GROUP\_2 in this table, group GROUP\_1 is on the left side of the selection screen, while group GROUP\_2 is on the right side.

Finally, you can adapt the properties of the parameters according to the requirements:

Parameters															
Name	Title	Group	Indented	Ma...	Hid...	Rea...	Enabled By Parameter	Screen Element	Radi...	Value ...	Value He...	Back...	Only Single Values	Text Lines in Editor	
NUMBERS_AVAILABLE	Numbers available	GROUP_1	No	No	No	No	NUMBERS_AVAILABLE	Checkbox	None	No	No	0		Add...	
NUMBERS	Numbers	GROUP_1	Yes	No	No	No		None	None	No	No	0		Edit...	
NAMES_AVAILABLE	Names available	GROUP_2	No	No	No	No	NAMES_AVAILABLE	Checkbox	None	No	No	0		Remove	
NAMES	Names	GROUP_2	Yes	No	No	No		None	None	No	Yes	0		Up	
TEXT	Some text	GROUP_3	No	No	No	No		None	None	No	No	4		Down	
														Synchronize	

The following parameter properties were changed in this example:

- The parameters NUMBERS\_AVAILABLE and NUMBERS were put into the first group of the first section (Group = GROUP\_1). Because parameter NUMBERS\_AVAILABLE is above parameter NUMBERS in this table, it's displayed above parameter NUMBERS on the selection screen. Parameter NUMBERS is output-indented on the selection screen (Indented = Yes). In addition, it's only active on the screen if the parameter NUMBERS\_AVAILABLE is set (Enabled By Parameter = NUMBERS\_AVAILABLE). Parameter NUMBERS\_AVAILABLE is displayed as checkbox on the selection screen (Screen Element = Checkbox).
- The parameters NAMES\_AVAILABLE and NAMES were put into another group (Group = GROUP\_2). The parameter NAMES is also output-indented (Indented = yes). In addition, it's only active on the selection screen if the parameter NAMES\_AVAILABLE is set (Enabled By Parameter = NAMES\_AVAILABLE). Parameter NAMES\_AVAILABLE is displayed as checkbox on the selection screen (Screen Element = Checkbox). Parameter NAMES only allows to enter multiple single values, but no value ranges (Only Single Values = Yes).
- The parameter TEXT is displayed in its own group (Group = GROUP\_3). It's displayed as text editor with four lines (Text Lines in Editor = 4).
- The parameter ANOTHER\_PUBLIC\_ATTRIBUTE is not relevant for the selection screen. Therefore, it was removed from the parameter list.

After you finished working on the catalog entry, don't forget to activate it.

After the application job catalog entry has been defined, you need to create an application job template. The template contains the parameter values that are displayed as default values on the selection screen. You can change these values before you schedule an application job. If you create a new template for your ABAP class, you may want that the parameters of the new template are already prefilled with default values. To determine these default values, the application job framework instantiates your class and calls the method `IF_APJ_DT_DEFAULTS~FILL_ATTRIBUTE_DEFAULTS`. It then reads the values of the class attributes and uses them as default values of the template parameters. These default values can either

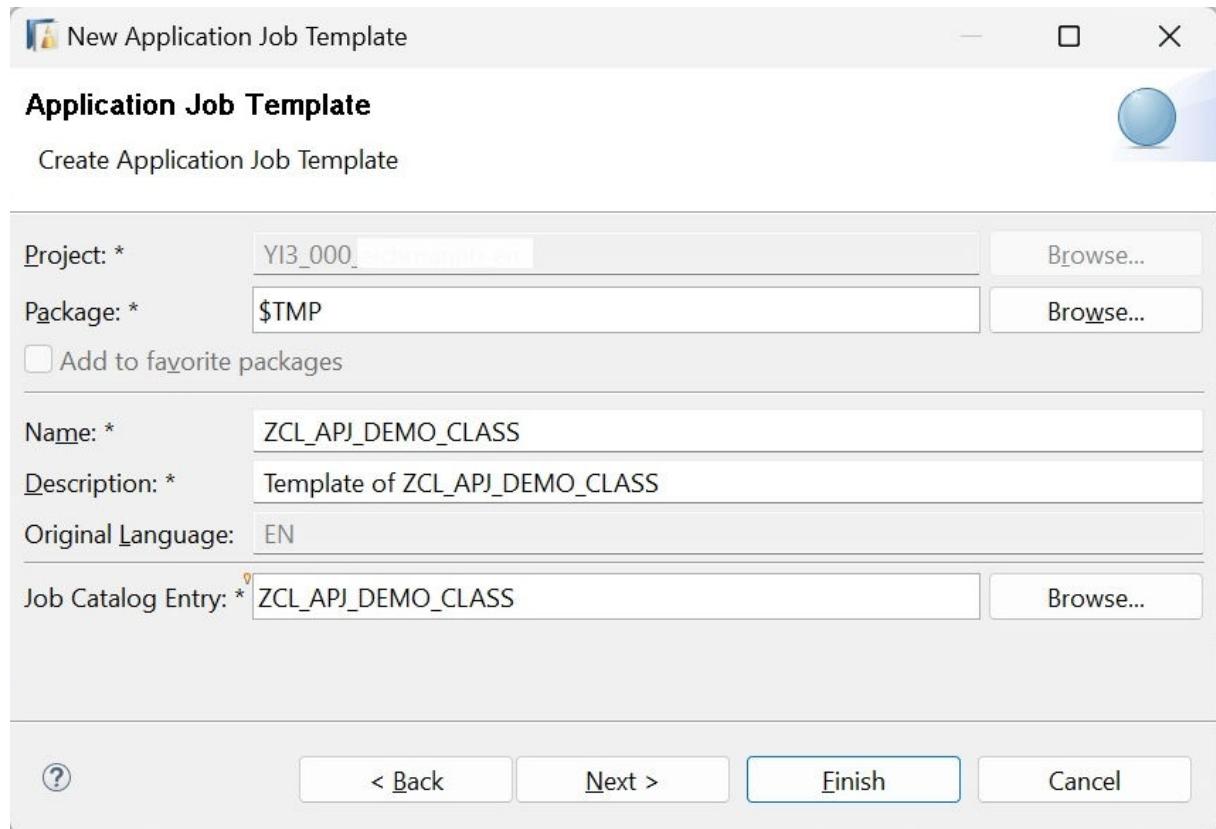
be set via the default values of the class attributes, via the CONSTRUCTOR of the class, or, if a more complex coding is required to determine these default values, you can also set them via the method `IF_APJ_DT_DEFAULTS~FILL_ATTRIBUTE_DEFAULTS`:

#### ↳ Sample Code

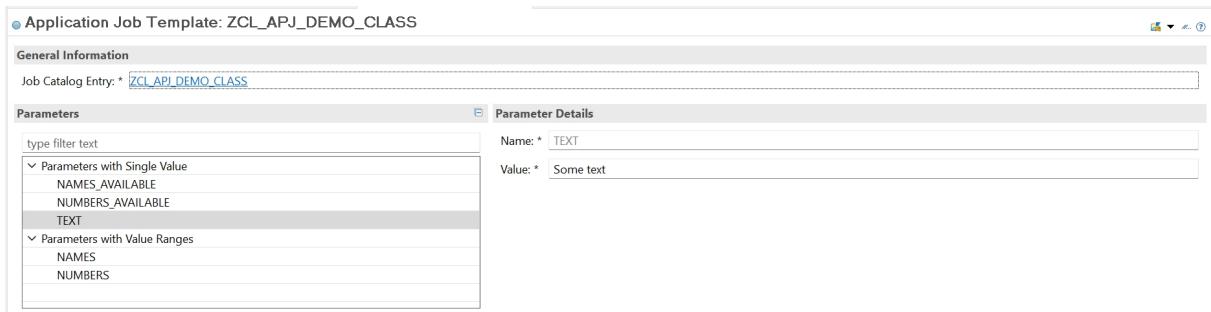
```
METHOD if_apj_dt_defaults~fill_attribute_defaults.  
    numbers = VALUE #( ( sign = 'I' option = 'BT' low = '1' high = '10' )  
                      ( sign = 'I' option = 'BT' low = '91' high = '100' ) ).  
    text = 'Some text'.  
ENDMETHOD.
```

Here, the default values of the attributes `NUMBERS` (which is a ranges table) and `TEXT` are set.

To create the new application job template, you can use the ABAP development tools for Eclipse. The template can be found under [Other ABAP Repository Object → Application Jobs → Application Job Template](#).



On the first screen, you select the name and a description text of the new template and the name of the catalog entry on which the template is based. It's the catalog entry which you just created.



After you've created the application job template, the parameters of the template are prefilled with the values which you set in the ABAP class. In this example, the following parameters got a default value:

- NUMBERS\_AVAILABLE and NAMES\_AVAILABLE are both set to x because these default values are set in the definition part of the class.
- NUMBERS is set to the ranges 1 - 10 and 91 - 100 and TEXT is set to the text Some text because of the implementation of method FILL\_ATTRIBUTE\_DEFAULTS.

Now, you can change the parameter values if you want. After you've finished this, don't forget to activate the template. With the new template, you can start to schedule an application job. If you do this, you get the following selection screen which meets the requirements set before:

## 4.2.11.8.1.4.1 Example Implementation Using Interfaces IF\_APJ\_DT\_EXEC\_OBJECT and IF\_APJ\_RT\_EXEC\_OBJECT

This example demonstrates the implementation of the concept of application jobs based on a class with the legacy interfaces IF\_APJ\_DT\_EXEC\_OBJECT and IF\_APJ\_RT\_EXEC\_OBJECT.

### **ⓘ Note**

Note that these are legacy interfaces. If you use these interfaces, you only have limited options during the editing of the job catalog entry. Don't use these interfaces for new developments, but use the interface IF\_APJ\_RT\_RUN instead. For more information, see [Example Implementation Using Interface IF\\_APJ\\_RT\\_RUN \[page 1510\]](#).

To implement the concept of application jobs, the ABAP class will contain the following attributes that will also be visible on the selection screen when the application job is scheduled:

- The attributes NUMBERS and NAMES which may contain value ranges
- The attribute TEXT which may contain a single value only
- The boolean attributes NUMBERS\_AVAILABLE and NAMES\_AVAILABLE. Attribute NUMBERS will only be used if NUMBERS\_AVAILABLE is set, while attribute NAMES will only be used if NAMES\_AVAILABLE is set.
- The parameter which belongs to NUMBERS\_AVAILABLE will be a checkbox.
- The parameter which belongs to NAMES\_AVAILABLE will be a checkbox.

Only limited options are available when interfaces `IF_APJ_RT_EXEC_OBJECT` and `IF_APJ_DT_EXEC_OBJECT` are used. For example, it is not possible to use the following features:

- Grouping of parameters on the selection screen, or putting them into sections.
- Controlling the enabled/disabled status of a parameter on the selection screen with another parameter.
- Restricting a field with value ranges to multiple single value, or setting the size of the text editor.

To implement these requirements, the ABAP class is created with the following definition part:

#### ↳ Sample Code

```
CLASS zcl_apj_demo_class2 DEFINITION PUBLIC FINAL CREATE PUBLIC.
  PUBLIC SECTION.
    INTERFACES if_apj_rt_exec_object.
    INTERFACES if_apj_dt_exec_object.

    TYPES:
      BEGIN OF ty_name_range,
        sign  TYPE c LENGTH 1,
        option  TYPE c LENGTH 2,
        low  TYPE c LENGTH 50,
        high  TYPE c LENGTH 50,
      END OF ty_name_range.
    TYPES: ty_name_ranges TYPE STANDARD TABLE OF ty_name_range WITH EMPTY KEY.
    !<p class="shorttext synchronized" lang="en">Numbers available</p>
    DATA numbers_available TYPE abap_bool VALUE abap_true.
    !<p class="shorttext synchronized" lang="en">Names available</p>
    DATA names_available TYPE abap_bool VALUE abap_true.
    !<p class="shorttext synchronized" lang="en">Names</p>
    DATA names TYPE ty_name_ranges.
    !<p class="shorttext synchronized" lang="en">Some text</p>
    DATA text TYPE c LENGTH 255.
    !<p class="shorttext synchronized" lang="en">Public attribute which is
    not used on the selection screen</p>
    DATA another_public_attribute TYPE c LENGTH 10.
  ENDCLASS.
```

The class definition contains the following parts:

- The class implements the interface `IF_APJ_RT_EXEC_OBJECT` and `IF_APJ_DT_EXEC_OBJECT`. This is necessary so that the class can be used in an application job.
- The class definition defines specific public attributes according to the requirements that will be set, these attributes are used by the logic of the class (or coding).

In the next step it is necessary to define the technical names and properties of all parameters that are used by all catalog entries and templates which are based on this class. To do this, the method GET\_PARAMETERS of interface IF\_APJ\_DT\_EXEC\_OBJECT must be implemented in the implementation part of the class:

#### ↳ Sample Code

```
METHOD if_apj_dt_exec_object~get_parameters.  
    et_parameter_def = VALUE #(  
        ( selname = 'NUM_AVAI' kind = if_apj_dt_exec_object=>parameter  
        datatype = 'C' length = 1 param_text = 'Numbers available' changeable_ind  
        = abap_true checkbox_ind = abap_true )  
        ( selname = 'NUMBERS' kind = if_apj_dt_exec_object=>select_option  
        datatype = 'I' param_text = 'Numbers' changeable_ind =  
        abap_true )  
        ( selname = 'NAM_AVAI' kind = if_apj_dt_exec_object=>parameter  
        datatype = 'C' length = 1 param_text = 'Names available' changeable_ind  
        = abap_true checkbox_ind = abap_true )  
        ( selname = 'NAMES' kind = if_apj_dt_exec_object=>select_option  
        datatype = 'C' length = 50 param_text = 'Names' changeable_ind  
        = abap_true )  
        ( selname = 'TEXT' kind = if_apj_dt_exec_object=>parameter  
        datatype = 'C' length = 255 param_text = 'Some text' changeable_ind  
        = abap_true )  
    ).  
ENDMETHOD.
```

In the method, you need to fill out an internal table with the list of all parameters to be used by the application job framework. The technical name of the parameters is SELNAME. Please notice:

- The length of the technical name is limited to 8 characters.
- The properties KIND, DATATYPE, and LENGTH should be defined according to the data type of the class attribute to which the parameter belongs.

The following parameter properties were set in this example:

- The titles of the parameters on the selection screen were set by property PARAM\_TEXT.
- The parameters NUM\_AVAI and NAM\_AVAI are displayed as a checkbox on the selection screen (checkbox\_ind = abap\_true).
- If any of the parameters are added to a catalog entry, the read-only property is set to false (changeable\_ind = abap\_true).
- The sequence of the parameters in table ET\_PARAMETER\_DEF defines the sequence of the parameters on the selection screen. For example, the parameter NUM\_AVAI is displayed above parameter NUMBERS.

In addition, in the implementation part of the class the method IF\_APJ\_RT\_EXEC\_OBJECT~EXECUTE must be implemented. This method is called by the application log framework when the application job is started. So, it should contain the business coding which will be run by the application job.

#### ↳ Sample Code

```
METHOD if_apj_rt_exec_object~execute.  
    LOOP AT it_parameters INTO DATA(l_parameter).  
        CASE l_parameter->selname.  
            WHEN 'NUM_AVAI'. numbers_available = l_parameter-low.  
            WHEN 'NUMBERS'.  
                APPEND VALUE #( sign = l_parameter-sign  
                option = l_parameter-option  
                low = l_parameter-low  
                high = l_parameter-high ) TO numbers.  
            WHEN 'NAM_AVAI'. names_available = l_parameter-low.
```

```

        WHEN 'NAMES'.
          APPEND VALUE #( sign    = l_parameter-sign
                          option   = l_parameter-option
                          low     = l_parameter-low
                          high    = l_parameter-high ) TO names.
        WHEN 'TEXT'. text = l_parameter-low.
      ENDCASE.
    ENDLOOP.
  TRY.
    DATA(l_log) = cl_bali_log=>create_with_header(
                  header = cl_bali_header_setter=>create( object =
'ZOBJECT'
                                         subobject =
'ZSUBOBJECT' ) .
      IF numbers_available = abap_true AND '42' IN numbers.
        l_log->add_item( item = cl_bali_free_text_setter=>create( severity
= if_bali_constants=>c_severity_information
                                         text =
'42 is in the number ranges' ) .
      ENDIF.
      IF names_available = abap_true AND names IS NOT INITIAL.
        l_log->add_item( item = cl_bali_free_text_setter=>create( severity
= if_bali_constants=>c_severity_status
                                         text =
'Some names are available' ) .
      ENDIF.
      l_log->add_item( item = cl_bali_free_text_setter=>create( severity =
if_bali_constants=>c_severity_status
                                         text = CONV
#( text ) ) .
      cl_bali_log_db=>get_instance( )->save_log_2nd_db_connection( log =
l_log
      assign_to_current_appl_job = abap_true .
      CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
        " some error handling
    ENDTRY.
  ENDMETHOD.

```

The values of the parameters that were set on the selection screen are available in the internal table `IT_PARAMETERS`. First, the table entries are read and the attributes of the class are filled in with the values. In this example, the global attributes of the class are filled in, you may also use local attributes of method `EXECUTE` to store the values. The field `SELNAME` of table `IT_PARAMETERS` contains the same technical names that you set in method `GET_PARAMETERS`.

In this simple example, the method `EXECUTE` creates an application log and assigns it to the current application job when the job is run. If the number 42 is in the ranges of field `NUMBERS`, a message is written into the log. Another message is written, if the field `NAMES` contains any entries. Finally, the content of the field `TEXT` is put into the log.

After you implemented the class, you should create a new catalog entry with ABAP development tools for Eclipse for this class so it can be used in an application job.

To do this, go to [Other ABAP Repository Object → Application Jobs → Application Job Template](#).

New Application Job Catalog Entry

### Application Job Catalog Entry

Create Application Job Catalog Entry

Project: \* YI3\_000\_

Package: \* \$TMP

Add to favorite packages

Name: \* ZCL\_API\_DEMO\_CLASS2

Description: \* Catalog Entry of ZCL\_API\_DEMO\_CLASS2

Original Language: EN

Class with Execute Method: \* ZCL\_API\_DEMO\_CLASS2

On the first screen you select the name and a description text of the new catalog entry and the name of the class which is run within the job. It shows the class that you just previously implemented.

After you have created the catalog entry, you can start to change it:

Name	Group	Indented	Mandatory	Hidden	Read Only	Enabled By Param...	Value Help	Value Help Type	Backend C...	Only Singl...	Text Lines in Editor
NUM_AVAI		No	No	No	No			None	No	No	0
NUMBERS		No	No	No	No			None	No	No	0
NAM_AVAI		No	No	No	No			None	No	No	0
NAMES		No	No	No	No			None	No	No	0
TEXT		No	No	No	No			None	No	No	0

Here, you can see that the parameter table was filled with all parameters which were defined in method GET\_PARAMETERS automatically. Please note that it is not possible to change the parameter properties here if the parameter properties were defined in the class via interface IF\_APJ\_DT\_EXEC\_OBJECT.

In the next step, you can optionally set Exit classes. After finishing your changes of the catalog entry, don't forget to activate it.

After the catalog entry has been defined, the next step is the creation of a template. The template contains the parameter values which are displayed as default values on the selection screen. As a user you may change these values before scheduling the application job. If you create a new template for your ABAP class, you might want the parameters of it to be pre-filled with default values already. You can set these default values in your ABAP class by filling out the internal table ET\_PARAMETER\_VAL in method GET\_PARAMETERS:

### ↔ Sample Code

```
METHOD if_apj_dt_exec_object~get_parameters.
  ...
    et_parameter_val = VALUE #(
      ( selname = 'NUM_AVAI' kind = if_apj_dt_exec_object->parameter sign
      = 'I' option = 'EQ' low = 'X' )
      ( selname = 'NUMBERS' kind = if_apj_dt_exec_object->select_option sign
      = 'I' option = 'BT' low = '1' high = '10' )
      ( selname = 'NUMBERS' kind = if_apj_dt_exec_object->select_option sign
      = 'I' option = 'BT' low = '91' high = '100' )
      ( selname = 'NAM_AVAI' kind = if_apj_dt_exec_object->parameter sign
      = 'I' option = 'EQ' low = 'X' )
      ( selname = 'TEXT' kind = if_apj_dt_exec_object->parameter sign
      = 'I' option = 'EQ' low = 'Some Text' )
    )).
ENDMETHOD.
```

Here, the default values of the attributes NUMBERS (which is a ranges table) and TEXT are set.

To create the new application job template, you can use the ABAP development tools for Eclipse. The template can be found under [Other ABAP Repository Object → Application Jobs → Application Job Template](#).

New Application Job Template

### Application Job Template

Create Application Job Template

**Project:** \* YI3\_000\_

**Package:** \* STMP

Add to favorite packages

**Name:** \* ZCL\_APJ\_DEMO\_CLASS2

**Description:** \* Template of ZCL\_APJ\_DEMO\_CLASS2

**Original Language:** EN

**Job Catalog Entry:** \*

On the first screen, you select the name and a description text of the new template and the name of the catalog entry on which the template is based. It's the catalog entry which you just created.

Application Job Template: ZCL\_APJ\_DEMO\_CLASS2

General Information

Job Catalog Entry: \* [ZCL\\_APJ\\_DEMO\\_CLASS2](#)

Parameters	Parameter Details
<input type="text" value="type filter text"/> <ul style="list-style-type: none"> <li>▼ Parameters with Single Value           <ul style="list-style-type: none"> <li>NAM_AVAI</li> <li>NUM_AVAI</li> <li>TEXT</li> </ul> </li> <li>▼ Parameters with Value Ranges           <ul style="list-style-type: none"> <li>NAME\$</li> <li>NUMBER\$</li> </ul> </li> </ul>	Name: * NAM_AVAI Value: * X

After you've created the application job template, the parameters of the template are prefilled with the values which you set in method GET\_PARAMETERS of the ABAP class.

Now, you can change the parameter values if you want. After you've finished this, don't forget to activate the template. With the new template, you can start to schedule an application job. If you do this, you get the following selection screen which meets the requirements set before:

#### 4.2.11.8.1.5 Creating a Job Catalog Entry and a Job Template via API

Find out how to create job catalog entries and job templates using a released API.

If you can't use the ABAP development tools for Eclipse to create job catalog entries and job templates, or if you have the requirement to create these objects via an ABAP class, you can use a released API. To use this API, you can use the ABAP class `CL_APJ_DT_CREATE_CONTENT`. The class offers the following methods:

- `GET_INSTANCE` (static): Get an instance of class `CL_APJ_DT_CREATE_CONTENT`
- `CREATE_JOB_CAT_ENTRY`: Create an application job catalog entry
- `CREATE_JOB_TEMPLATE_ENTRY`: Create an application job template
- `DELETE_JOB_CAT_ENTRY`: Delete an application job catalog entry
- `DELETE_JOB_TEMPLATE_ENTRY`: Delete an application job template
- `EXISTS_JOB_CAT_ENTRY`: Check the existence of an application job catalog entry
- `EXISTS_JOB_TEMPLATE_ENTRY`: Check the existence of an application job template

The following code sample generates one job catalog entry and one related job template:

##### Sample Code

```
CLASS zcl_test_apj_simple_obj_gen DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_test_apj_simple_obj_gen IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    CONSTANTS lc_catalog_name      TYPE
    cl_apj_dt_create_content=>ty_catalog_name  VALUE 'ZTEST_MY_SIMPLE_JOB'.
    CONSTANTS lc_catalog_text      TYPE
    cl_apj_dt_create_content=>ty_text          VALUE 'My first simple application
job'.
    CONSTANTS lc_class_name        TYPE
    cl_apj_dt_create_content=>ty_class_name    VALUE 'ZCL_TEST_APJ_SIMPLE'.
    CONSTANTS lc_template_name     TYPE
    cl_apj_dt_create_content=>ty_template_name VALUE 'ZTEST_MY_SIMPLE_JOB_TEMPL'.
```

```

    CONSTANTS lc_template_text      TYPE
    cl_apj_dt_create_content=>ty_text          VALUE 'My first simple job
template'.
    CONSTANTS lc_transport_request TYPE
    cl_apj_dt_create_content=>ty_transport_request VALUE 'Y11K900361'.
    CONSTANTS lc_package           TYPE
    cl_apj_dt_create_content=>ty_package        VALUE 'Z_D028092_APJ_MAIN'.
    DATA(lo_dt) = cl_apj_dt_create_content=>get_instance( ).
    " Create job catalog entry (corresponds to the former report incl.
selection parameters)
    " Provided implementation class iv_class_name shall implement two
interfaces:
    " - if_apj_dt_exec_object to provide the definition of all supported
selection parameters of the job
    " (corresponds to the former report selection parameters) and to
provide the actual default values
    " - if_apj_rt_exec_object to implement the job execution
TRY.
    lo_dt->create_job_cat_entry(
        iv_catalog_name      = lc_catalog_name
        iv_class_name        = lc_class_name
        iv_text              = lc_catalog_text
        iv_catalog_entry_type = cl_apj_dt_create_content=>class_based
        iv_transport_request = lc_transport_request
        iv_package            = lc_package
    ).
    out->write( |Job catalog entry created successfully| ).
    CATCH cx_apj_dt_content INTO DATA(lx_apj_dt_content).
        out->write( |Creation of job catalog entry failed:
{ lx_apj_dt_content->get_text( ) }| ).
    ENDTRY.
    " Create job template (corresponds to the former system selection
variant) which is mandatory
    " to select the job later on in the Fiori app to schedule the job
    DATA lt_parameters TYPE if_apj_dt_exec_object=>tt_templ_val.
    NEW zcl_test_apj_simple( )->if_apj_dt_exec_object~get_parameters(
        IMPORTING
            et_parameter_val = lt_parameters
    ).
TRY.
    lo_dt->create_job_template_entry(
        iv_template_name     = lc_template_name
        iv_catalog_name      = lc_catalog_name
        iv_text              = lc_template_text
        it_parameters        = lt_parameters
        iv_transport_request = lc_transport_request
        iv_package            = lc_package
    ).
    out->write( |Job template created successfully| ).
    CATCH cx_apj_dt_content INTO lx_apj_dt_content.
        out->write( |Creation of job template failed: { lx_apj_dt_content-
>get_text( ) }| ).
    RETURN.
ENDTRY.
ENDMETHOD.
ENDCLASS.
```

## 4.2.11.8.1.6 Setting up the Authorizations

Some further activities in ABAP development tools for Eclipse and in the administrator's launchpad are necessary to be able to schedule the job template in the Fiori app *Application Jobs*.

1. Create an Identity and Access Management (IAM) business catalog.

When you create a job catalog entry and a job template as explained in [Creating a Job Catalog Entry and a Job Template \[page 1502\]](#), an object of the type **IAM App** is created automatically. It has the name <job catalog entry name>\_SAJC. This IAM app contains the start authorization for all job templates that refer to this job catalog entry.

Create an IAM business catalog via ABAP development tools for Eclipse:

1. Right-click the package where the objects are located that have already been created.
  2. Select *New > Other ABAP Repository Object*.
  3. Expand *Cloud Identity and Access Management*.
  4. Double-click *Business Catalog*.
  5. Enter a name (such as ZTEST\_MY\_SIMPLE\_JOBS) and a description.
  6. Select *Save* and choose the appropriate transport request.
  7. Select the tab *Apps* below.
  8. Press *Add ...* on the top right corner of the screen.
  9. Enter the name of the IAM app mentioned in step e (ZTEST\_MY\_SIMPLE\_JOB\_SAJC).
  10. Select *Save*.
  11. Press *Publish Locally* on the top right corner of the screen.
2. If the business logic performs own authorization checks, remember to add the required authorizations to a separate IAM app of type **EXT**. The following steps explain how to do this:
    1. Right-click the package that contains the objects that have already been created.
    2. Select *New* and then choose *Other ABAP Repository Object*.
    3. Expand *Cloud Identity and Access Management*.
    4. Choose *IAM App*.
    5. Enter a name and a description. Choose *EXT - External App* as application type.
    6. Select an appropriate transport request and click *Save*.
    7. Select the *Authorizations* tab.
    8. Select *Insert*. Now, you can specify an authorization object and assign values to the authorization fields.
    9. Repeat step h for each authorization that is necessary to run the business logic.
  10. Add the IAM app to the business catalog that you've created as described in step 1.

3. Create a business role.

Open the Fiori app *Maintain Business Roles* in the Fiori launchpad of the administrator and perform the following steps:

1. Click *New*.
2. Choose a name (such as ZTEST\_MY\_SIMPLE\_JOBS) and a description.
3. Click on *Assigned Business Catalogs*.
4. Select *Add*.
5. Select the business catalog created in the previous step (such as ZTEST\_MY\_SIMPLE\_JOBS).

### ⓘ Note

A business user scheduling application jobs needs to have access to the *Application Jobs* tile. This tile is provided by the business catalog SAP\_CORE\_BC\_APJ\_JCE, which is contained in the

business role SAP\_BR\_ADMINISTRATOR. It doesn't only provide the *Application Jobs* app, but also contains restriction types to configure the authorization on other users' jobs and to schedule jobs for other users (for more information, see the section *Assigning Further Authorizations* below).

Add the business catalog SAP\_CORE\_BC\_APJ\_JCE to the business role created following the steps above, or assign it to a separate business role.

6. Select *Save*.
4. Assign the business role to a user.

For more information, see [How to Maintain Business Users \[page 2656\]](#).

## Assigning Further Authorizations

You can access only your own jobs in the *Application Jobs* app. In order to grant access to other users' objects used in the *Application Jobs* app, create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_JCE to it. You can modify the provided restriction types according to your needs.

1. Create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_JCE to it. The restriction type *Job Catalog Entry/Application Job Part* contains the two fields *Application Job Catalog Entry* and *Application Job Part*.

### ⓘ Note

With this restriction type you'll be able to provide the necessary authorizations for certain actions on other users' application jobs, which are based on a specified job catalog entry. With this, you can display the logs of other users' jobs, which are based on this specified job catalog entry.

Alternatively, you can also create a display only-role using this restriction type.

2. The business catalog SAP\_CORE\_BC\_APJ\_JCE has another restriction type: *Create Application Jobs for Other Users*. It has the restriction field *Application Job Catalog Entry*.

### ⓘ Note

With this restriction type you'll be able to grant a business user the authorization to schedule an application job for any other business user, based on the job catalog entry selected. This feature is supported by the parameter *iv\_username* of the method *c1\_apj\_rt\_api=>schedule\_job*. For more information, see [Maintaining Application Jobs Using an API \[page 1527\]](#).

3. Next, define your business role with the above-mentioned restriction fields *Application Job Catalog Entry* and *Application Job Part*. To do so, select *Maintain Restrictions*. In the *Application Job Catalog Entry* field, you can choose the job catalog entries, and in the *Application Job Part* field, you can choose for which part of a job access should be granted. You can choose the values **JOB** (*Job Details*), **SPOOL** (*Result List*), and **APLOG** (*Log*) by clicking the *Edit* icon next to the fields.
4. Go back and select *Save*. Now, you've maintained your business role, and you can assign the role to a business user.

## 4.2.11.8.1.7 Deleting the Job Catalog Entry and the Job Template

Find out how to delete a job catalog entry and a related job template.

To delete a job catalog entry or a job template, use the ABAP development tools for Eclipse. Right-click on the object in the *Project Explorer* and select *Delete*.

Alternatively, you could also still delete job catalog entries and job templates via a released API. For more information, see [Creating a Job Catalog Entry and a Job Template via API \[page 1523\]](#).

Before you can delete a job catalog entry, you must first delete all related job templates: When you create a job catalog entry, an IAM app that is based on the job catalog entry is automatically created in the Cloud. If you have followed the steps in [Setting up the Authorizations \[page 1525\]](#), you should perform additional steps before deleting the job catalog entry, since otherwise, the IAM app can't automatically be deleted together with the job catalog entry:

- Remove the business role from the business user in the Fiori app [Maintain Business Users](#)
- Delete the business role in the Fiori app [Maintain Business Roles](#)
- Unassign the IAM app from the IAM business catalog in ABAP development tools for Eclipse
- Delete the IAM business catalog in ABAP development tools for Eclipse

Now, you can delete the job catalog entry.

If you only want to replace the job catalog entry (if you want to delete it to recreate it with the same name), it's sufficient to remove the IAM app assignment from the IAM business catalogs. After the creation of the new job catalog entry, assign the new IAM app to the IAM business catalogs.

## 4.2.11.8.2 Maintaining Application Jobs Using an API

You can use the `CL_APJ_RT_API` class to maintain application jobs. You can do the following:

- schedule an application job
- retrieve the status of an application job
- cancel an application job
- delete an application job

### ⌚ Example

#### ↔ Sample Code

```
class z_cl_rt_api_demo definition
public
final
create public .
public section.
    interfaces if_oo_adt_classrun.
protected section.
private section.
endclass.
class z_cl_rt_api_demo implementation.
```

```

"
<SIGNATURE>-----
-----+
" | Instance Public Method Z_CL_RT_API_DEMO->IF_OO_ADT_CLASSRUN~MAIN
"
+-----+
-----+
" | [--->] OUT                                     TYPE REF TO IF_OO_ADT_CLASSRUN_OUT
"
+-----+
-----</SIGNATURE>
method if_oo_adt_classrun~main.
    data lv_job_text type cl_apj_rt_api=>ty_job_text value 'Demo_Job'.
    data lv_template_name type cl_apj_rt_api=>ty_template_name.
    data ls_start_info type cl_apj_rt_api=>ty_start_info.
    data ls_scheduling_info type cl_apj_rt_api=>ty_scheduling_info.
    data ls_end_info type cl_apj_rt_api=>ty_end_info.
    data lt_job_parameters type cl_apj_rt_api=>tt_job_parameter_value.
    data ls_job_parameters type cl_apj_rt_api=>ty_job_parameter_value.
    data ls_value type cl_apj_rt_api=>ty_value_range.
    data lv_jobname type cl_apj_rt_api=>ty_jobname.
    data lv_jobcount type cl_apj_rt_api=>ty_jobcount.
    data lv_status type cl_apj_rt_api=>ty_job_status.
    data lv_statustext type cl_apj_rt_api=>ty_job_status_text.
    data lv_txt type string.
    data ls_ret type bapiret2.

" Choose the name of the existing job template.
    lv_template_name = 'ZTEST_MY_SIMPLE_JOB_TEMP'.
" The immediate start can't be used when being called from within a RAP
business object
" because the underlying API performs an implicit COMMIT WORK.
"     ls_start_info-start_immediately = 'X'.
" Start the job using a timestamp instead. This will start the job
immediately but can have a delay depending on the current workload.
    get time stamp field data(ls_ts1).
" Add 1 hour
    data(ls_ts2) = cl_abap_tstmp=>add( tstmp = ls_ts1
                                         secs   = 3600 ).
    ls_start_info-timestamp = ls_ts2.

" Periodicity
    ls_scheduling_info-periodic_granularity = 'D'.
    ls_scheduling_info-periodic_value = 1.
    ls_scheduling_info-test_mode = abap_false.
    ls_scheduling_info-timezone = 'CET'.
    ls_end_info-type = 'NUM'.
    ls_end_info-max_iterations = 3.

" Fill parameter table:
" Fill the table only if you want to overrule the parameter values
" which are stored in the template.
" The field names in this program must match the field names of the
template.
    ls_job_parameters-name = 'P_TEST1'.
    ls_value-sign = 'I'.
    ls_value-option = 'EQ'.
    ls_value-low = 'Test 1'.
    append ls_value to ls_job_parameters-t_value.
    append ls_job_parameters to lt_job_parameters.
    clear ls_job_parameters.
    ls_job_parameters-name = 'P_TEST2'.
    ls_value-sign = 'I'.
    ls_value-option = 'BT'.
    ls_value-low = 'ATEST'.
    ls_value-high = 'ZTEST'.
    append ls_value to ls_job_parameters-t_value.
    ls_job_parameters-name = 'P_TEST2'.
    ls_value-sign = 'I'.
    ls_value-option = 'BT'.
    ls_value-low = '11111'.

```

```

ls_value-high = '99999'.
append ls_value to ls_job_parameters-t_value.
append ls_job_parameters to lt_job_parameters.
clear ls_job_parameters.
ls_job_parameters-name = 'P_TEST3'.
ls_value-sign = 'I'.
ls_value-option = 'EQ'.
ls_value-low = '220'.
append ls_value to ls_job_parameters-t_value.
append ls_job_parameters to lt_job_parameters.
clear ls_job_parameters.
try.
" Some scenarios require that the job key ( = jobname, jobcount) is already
known
" before the job is created. The method generate_jobkey creates a valid job
key.
" This key can then be passed later on to the method schedule_job, and a
job with
" exactly this key is created.
" Optional. You need this call only if you have to know the job key in
advance.
"         cl_apj_rt_api=>generate_jobkey(
"             importing
"                 ev_jobname    = lv_jobname
"                 ev_jobcount   = lv_jobcount ).
" If you pass the table lt_job_parameters , then the parameters
" contained in this table are used.
" If you don't pass the table, the parameters contained in the
" job template are used.
        cl_apj_rt_api=>schedule_job(
            exporting
                iv_job_template_name   = lv_template_name
                iv_job_text           = lv_job_text
                is_start_info          = ls_start_info
                is_scheduling_info     = ls_scheduling_info
                is_end_info            = ls_end_info
                it_job_parameter_value = lt_job_parameters
" The following two parameters are optional. If you pass them, they must
have been generated
" with the optional call generate_jobkey above.
"         iv_jobname          = lv_jobname
"         iv_jobcount         = lv_jobcount
            importing
                ev_jobname          = lv_jobname
                ev_jobcount         = lv_jobcount
).
out->write( lv_jobname ).
out->write( lv_jobcount ).
cl_apj_rt_api=>get_job_status(
            exporting
                iv_jobname          = lv_jobname
                iv_jobcount         = lv_jobcount
            importing
                ev_job_status        = lv_status
                ev_job_status_text   = lv_statustext
).
out->write( lv_status ).
out->write( lv_statustext ).

" Via the following method you can cancel the job
" in the application job context 'cancel' means (as in the Fiori app):
" 1. if the job is running, it will be canceled
" 2. if the job has not yet started, it will be deleted.
" In case the job is periodic, the whole periodicity chain is deleted.
        cl_apj_rt_api=>cancel_job(
            exporting
                iv_jobname          = lv_jobname
                iv_jobcount         = lv_jobcount
).

```

```

        catch cx_apj_rt into data(exc).
        lv_txt = exc->get_longtext( ).
        ls_ret = exc->get_bapiret2( ).
        out->write( 'ERROR:' ).
        out->write( lv_txt ).
        out->write( 'msg type =' ).
        out->write( ls_ret-type ).
        out->write( 'msg id =' ).
        out->write( ls_ret-id ).
        out->write( 'msg number =' ).
        out->write( ls_ret-number ).
        out->write( 'msg message =' ).
        out->write( ls_ret-message ).

    endtry.
endmethod.
endclass.

```

#### 4.2.11.8.2.1 Examples of Periodic Application Jobs

You can schedule application jobs to run periodically.

Here, you'll find a few codesamples that specify various periodicities. These code samples use the new human-readable constants of the class CL\_APJ\_RT\_API. Let's start with a codesample showing an application job that runs monthly, always on the second day of the month:

##### ↔ Sample Code

```

CLASS zcl_apj_monthly DEFINITION
PUBLIC
FINAL
CREATE PUBLIC .
PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
PROTECTED SECTION.
PRIVATE SECTION.
ENDCLASS.
CLASS zcl_apj_monthly IMPLEMENTATION.
"
<SIGNATURE>-----
-----+
" | Instance Public Method ZCL_APJ_MONTHLY->IF_OO_ADT_CLASSRUN~MAIN
"
+-----+
-----+
" | [--->] OUT                                     TYPE REF TO IF_OO_ADT_CLASSRUN_OUT
"
+-----+
-----</SIGNATURE>
METHOD if_oo_adt_classrun~main.
" the following coding creates an application job with the following start
condition:
"
" periodicity: every month on the second day - starting September 2nd, 2024
" first start: specified by variable ls_start_info-timestamp
"
" Exception rule: If the execution day is a non-working day according to a
"                   specified calendar, the job should be skipped.
"
" End of series: after 10 executions - specified by ls_end_info

```

```

DATA job_text          TYPE cl_apj_rt_api=>ty_job_text VALUE
'Monthly_Job'.
DATA template_name    TYPE cl_apj_rt_api=>ty_template_name.
DATA ls_start_info     TYPE cl_apj_rt_api=>ty_start_info.
DATA ls_scheduling_info TYPE cl_apj_rt_api=>ty_scheduling_info.
DATA ls_job_exception  TYPE cl_apj_rt_api=>ty_job_exception.
DATA ls_end_info        TYPE cl_apj_rt_api=>ty_end_info.
DATA jobname           TYPE cl_apj_rt_api=>ty_jobname.
DATA jobcount          TYPE cl_apj_rt_api=>ty_jobcount.
DATA status             TYPE cl_apj_rt_api=>ty_job_status.
DATA statustext         TYPE cl_apj_rt_api=>ty_job_status_text.
DATA: txt TYPE string.
DATA: tz  TYPE timezone.
DATA dat TYPE d.
DATA tim TYPE t.
template_name = 'ZTEST_MY_SIMPLE_JOB_TEMP'.
" the immediate start would look like this:
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.
  dat = '20240902'.
  tim = '040000'.
  tz = cl_abap_tstmp=>get_system_timezone(   ).
  CONVERT DATE dat TIME tim
    INTO TIME STAMP DATA(ts) TIME ZONE tz.
  ls_start_info-timestamp = ts.
" periodicity
  ls_scheduling_info-periodic_granularity  = cl_apj_rt_api=>period_months.
" monthly
  ls_scheduling_info-periodic_value        = 1.
  ls_scheduling_info-test_mode            = abap_false.
  ls_scheduling_info-timezone            = tz.
" exception rule
  ls_job_exception-calender_id = '01'.
  ls_job_exception-start_restriction_code =
cl_apj_rt_api=>dont_process_on_holiday. "skip, if the execution day is a
holiday
  ls_scheduling_info-exception = ls_job_exception.
" the periodic job should not run anymore after the 10th execution
  ls_end_info-type = cl_apj_rt_api=>period_end_by_nr_of_executions.
  ls_end_info-max_iterations = 10.
TRY.
  cl_apj_rt_api=>schedule_job(
    EXPORTING
      iv_job_template_name = template_name
      iv_job_text          = job_text
      is_start_info         = ls_start_info
      is_scheduling_info   = ls_scheduling_info
      is_end_info           = ls_end_info
      iv_jobname            = jobname
      iv_jobcount           = jobcount
    IMPORTING
      ev_jobname            = jobname
      ev_jobcount           = jobcount
  ).
  CONCATENATE jobname jobcount INTO txt SEPARATED BY space.
  out->write( txt ).
  cl_apj_rt_api=>get_job_status(
    EXPORTING
      iv_jobname            = jobname
      iv_jobcount           = jobcount
    IMPORTING
      ev_job_status          = status
      ev_job_status_text     = statustext
  ).
  CONCATENATE 'status =' statustext INTO txt SEPARATED BY space.
  out->write( txt ).
CATCH cx_apj_rt INTO DATA(exc).
  txt = exc->get_longtext(   ).
  out->write( txt ).

```

```
ENDTRY.  
ENDMETHOD.  
ENDCLASS.
```

You can specify more periodicities with the method `CL_APJ_RT_API=>SCHEDULE_JOB`. The code snippets below show and explain in comments how to fill the method's parameters `IS_START_INFO`, `IS_SCHEDULING_INFO` (including the substructures `EXCEPTION`, `WEEKDAY_INFO`, and `MONTH_INFO`), and `IS_END_INFO`.

### ⓘ Note

Note that there are hard-coded dates and times in the examples.

Each of the examples needs the following data definitions (or at least a subset):

### ↴ Sample Code

```
DATA ls_start_info      TYPE cl_apj_rt_api=>ty_start_info. "for method  
parameter IS_START_INFO  
DATA ls_scheduling_info TYPE cl_apj_rt_api=>ty_scheduling_info. "for method  
parameter IS_SCHEDULING_INFO  
DATA ls_job_exception  TYPE cl_apj_rt_api=>ty_job_exception. "substructure  
of ls_scheduling_info  
DATA ls_weekday_info   TYPE cl_apj_rt_api=>ty_weekday_info. "substructure  
of ls_scheduling_info  
DATA ls_month_info     TYPE cl_apj_rt_api=>ty_month_info. "substructure of  
ls_scheduling_info  
DATA ls_end_info       TYPE cl_apj_rt_api=>ty_end_info. "for method  
parameter IS_END_INFO  
DATA tz    TYPE timezone.  
DATA dat  TYPE d.  
DATA tim  TYPE t.  
DATA ts   TYPE timestamp.
```

### Example 1

This code snippet shows how to specify the start condition of the periodicity of the fourth from the last day every month, starting from 28 January, 2024. Its first start is specified by the variable `LS_START_TIMESTAMP`. The application job stops running after ten executions, specified by `LS_END_INFO`.

### ↴ Sample Code

```
" the immediate start would look like this:  
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.  
dat = '20241028'. "should be the exact date of the first execution  
tim = '040000'.  
tz = cl_abap_tstmp=>get_system_timezone( ).  
CONVERT DATE dat TIME tim  
      INTO TIME STAMP ts TIME ZONE tz.  
ls_start_info-timestamp = ts.  
" periodicity  
ls_scheduling_info-periodic_granularity = cl_apj_rt_api=>period_months.  
ls_scheduling_info-periodic_value      = 1.  
ls_scheduling_info-test_mode          = abap_false.  
ls_scheduling_info-timezone          = tz.  
ls_month_info-day = 4.  
ls_month_info-shift_direction = cl_apj_rt_api=>from_end_of_month.  
" there is also the constant  
" cl_apj_rt_api=>FROM_BEGINNING_OF_MONTH  
ls_scheduling_info-month_info      = ls_month_info.  
" the periodic job should not run anymore after the 10th execution
```

```
ls_end_info-type = cl_apj_rt_api=>period_end_by_nr_of_executions.  
ls_end_info-max_iterations = 10.
```

## Example 2

This code snippet shows how to specify the start condition of the periodicity of the second day of every month, starting from 2 September, 2024. Its first start is specified by the variable LS\_START\_INFO-TIMESTAMP. The exception rule is that if the execution day is a non-working day according to a specified calendar, the job should be skipped. The application job stops running after ten executions, specified by LS\_END\_INFO.

### ↔ Sample Code

```
" the immediate start would look like this:  
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.  
dat = '20240902'.  
tim = '040000'.  
tz = cl_abap_tstmp=>get_system_timezone( ).  
CONVERT DATE dat TIME tim  
    INTO TIME STAMP ts TIME ZONE tz.  
ls_start_info-timestamp = ts.  
" periodicity  
ls_scheduling_info-periodic_granularity      = cl_apj_rt_api=>period_months. "  
monthly  
ls_scheduling_info-periodic_value           = 1.  
ls_scheduling_info-test_mode                = abap_false.  
ls_scheduling_info-timezone                = tz.  
" exception rule  
ls_job_exception-calender_id = '01'.  
ls_job_exception-start_restriction_code =  
cl_apj_rt_api=>dont_process_on_holiday. "skip if the execution day is a  
holiday  
" There are also the following constants for exception rules (names are self-  
explanatory):  
" cl_apj_rt_api=>PROCESS_BEFORE_HOLIDAY  
" cl_apj_rt_api=>PROCESS_AFTER_HOLIDAY  
" cl_apj_rt_api=>PROCESS_ALWAYS  
ls_scheduling_info-exception = ls_job_exception.  
" the periodic job should not run anymore after the 10th execution  
ls_end_info-type = cl_apj_rt_api=>period_end_by_nr_of_executions.  
ls_end_info-max_iterations = 10.
```

## Example 3

This code snippet shows how to specify the start condition of the periodicity of every week on Monday and Thursday. Its first start is specified by the variable LS\_START\_INFO-TIMESTAMP. The application job stops running after a specific time, specified by LS\_END\_INFO-TIMESTAMP.

### ⓘ Note

The application job start in this code snippet has to match a valid start date (a Monday or a Thursday).

### ↔ Sample Code

```
" the immediate start would look like this  
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.  
" In the following the assignment  
"     dat  
" must be a Monday or a Thursday, because the job should run  
" every week on Monday and Thursday.  
" Otherwise, an error will be reported.  
" So the variable
```

```

" ls_start_info-timestamp
" will be the start date and time of the series.
dat = '20240219'.
tim = '180000'.
tz = cl_abap_tstmp=>get_system_timezone(  ).
CONVERT DATE dat TIME tim
    INTO TIME STAMP ts TIME ZONE tz.
ls_start_info-timestamp = ts.
" periodicity
" the job should run every third Tuesday of the month
ls_weekday_info-on_monday      = abap_true.
ls_weekday_info-on_thursday    = abap_true.
ls_scheduling_info-periodic_granularity = cl_apj_rt_api=>period_weeks. "
weekly
ls_scheduling_info-periodic_value      = 1.
ls_scheduling_info-test_mode          = abap_false.
ls_scheduling_info-timezone           = tz.
ls_scheduling_info-weekday_info       = ls_weekday_info.
" the periodic job should not run anymore after Dec. 31st 2030, 11:00 time
zone tz (system time zone)
ls_end_info-type = cl_apj_rt_api=>period_end_by_date.
dat = '20300331'.
tim = '110000'.
CONVERT DATE dat TIME tim
    INTO TIME STAMP ls_end_info-timestamp TIME ZONE tz.

```

#### Example 4

This code snippet shows how to specify the start condition of the periodicity of every week on Friday. Its first start is specified by the variable LS\_START\_INFO-TIMESTAMP. The exception rule is that if a Friday is a non-working day according to a specified calendar, the job should run on the next working day. The application job stops running at a specific day, specified by LS\_END\_INFO-TIMESTAMP.

#### ⓘ Note

The application job start in this code snippet has to match a valid start date (a Friday).

#### ↪ Sample Code

```

" the immediate start would look like this
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.
" In the following the assignment
"     dat
" must be a Friday, because the job should run
" every week on Friday.
" Otherwise, an error will be reported.
" So the variable
"     ls_start_info-timestamp
" will be the start date and time of the series.
dat = '20240823'.
tim = '040000'.
" alternatively, we can work with the system timezone
" tz = CL_ABAP_TSTMP=>get_system_timezone(  ).
CONVERT DATE dat TIME tim
    INTO TIME STAMP ts TIME ZONE 'CET'. "means: the first job should
start at 4:00 CET
ls_start_info-timestamp = ts.
" periodicity
ls_scheduling_info-periodic_granularity = cl_apj_rt_api=>period_weeks. "
weekly
ls_scheduling_info-periodic_value      = 1.
ls_scheduling_info-test_mode          = abap_false.
" Below, the timezone for the periodicity is set. In combination with the
above first start time 4:00

```

```

" the periodicity time zone CET means: The job should always start at 4:00
CET - no matter if daylight savings time
" is active or not.
" If we don't set the periodicity timezone, it will be set to the system
timezone by default. Let's assume UTC.
" Since the first start date and time of the job is 20240223 040000 CET, it
is 20240223 030000 UTC.
" The job will then run at 3:00 UTC every Friday, which is 4:00 CET in
'winter' and 5:00 CET in 'summer'.
ls_scheduling_info-timezone          = 'CET'.
" the job should run every Friday
ls_weekday_info-on_friday        = abap_true.
ls_scheduling_info-weekday_info = ls_weekday_info.
" exception rule
ls_job_exception-calender_id = '01'.
ls_job_exception-start_restriction_code =
cl_apj_rt_api=>process_after_holiday. "run on the next working day
ls_scheduling_info-exception = ls_job_exception.
" the periodic job should not run anymore after Dec. 31st 2030, 11:00 time
zone tz (system time zone)
ls_end_info-type = cl_apj_rt_api=>period_end_by_date.
dat = '20300331'.
tim = '110000'.
CONVERT DATE dat TIME tim
    INTO TIME STAMP ls_end_info-timestamp TIME ZONE tz.

```

### Example 5

This code snippet shows how to specify the start condition of the periodicity of the second Thursday of every month. Its first start is specified by the variable LS\_START\_INFO-TIMESTAMP. The application job stops running after 10 executions, specified by LS\_END\_INFO.

#### Sample Code

```

" the immediate start would look like this:
" ls_start_info-start_immediately = cl_apj_rt_api=>start_immediately.
" start date and time of series
dat = '20240208'. " this date must match the date of the first job execution
tim = '180000'.
tz = cl_abap_tstmp=>get_system_timezone( ).
CONVERT DATE dat TIME tim
    INTO TIME STAMP ts TIME ZONE tz.
ls_start_info-timestamp = ts.
" periodicity
ls_scheduling_info-periodic_granularity      =
cl_apj_rt_api=>period_weekday_in_month.
ls_scheduling_info-periodic_value            = 1.
ls_scheduling_info-test_mode                = abap_false.
ls_scheduling_info-timezone                 = tz.
ls_weekday_info-on_thursday                = abap_true.
ls_scheduling_info-weekday_info = ls_weekday_info.
ls_month_info-week_number = 2.
ls_scheduling_info-month_info   = ls_month_info.
" the periodic job should not run anymore after the 10th execution
ls_end_info-type = cl_apj_rt_api=>period_end_by_nr_of_executions.
ls_end_info-max_iterations = 10.

```

### New Constants of the Class CL\_APJ\_RT\_API:

For periodicity:

- CL\_APJ\_RT\_API=>PERIOD\_MINUTES
- CL\_APJ\_RT\_API=>PERIOD\_HOURS

- CL\_APJ\_RT\_API=>PERIOD\_DAYS
- CL\_APJ\_RT\_API=>PERIOD\_WEEKS
- CL\_APJ\_RT\_API=>PERIOD\_MONTHS
- CL\_APJ\_RT\_API=>PERIOD\_WEEKDAY\_IN\_MONTH (see example 5)

For immediate start:

- CL\_APJ\_RT\_API=>START\_IMMEDIATELY (used in all above examples, but commented out)

For end of condition series:

- CL\_APJ\_RT\_API=>PERIOD\_END\_BY\_DATE (see example 5, for example)
- CL\_APJ\_RT\_API=>PERIOD\_END\_BY\_NR\_OF\_EXECUTIONS (see example 2, for example)

For count direction (from the start or end of the month):

- CL\_APJ\_RT\_API=>FROM\_BEGINNING\_OF\_MONTH (see example 1)
- CL\_APJ\_RT\_API=>FROM\_END\_OF\_MONTH (see example 1)

Exception rules (how to treat an application job if the regular job execution is (or would be) on a holiday according to the specified calendar):

- CL\_APJ\_RT\_API=>DONT\_PROCESS\_ON\_HOLIDAY
- CL\_APJ\_RT\_API=>DONT\_PROCESS\_BEFORE\_HOLIDAY (for example, on the last working day before the holiday)
- CL\_APJ\_RT\_API=>PROCESS\_AFTER\_HOLIDAY (for example, on the first working day after the holiday)
- CL\_APJ\_RT\_API=>PROCESS\_ALWAYS

Only relevant for the method CL\_APJ\_RT\_API=>RESTART\_JOB:

- CL\_APJ\_RT\_API=>RESTART\_FROM\_ERROR\_STEP
- CL\_APJ\_RT\_API=>RESTART\_AFTER\_ERROR\_STEP

## 4.2.11.9 Application Logs

You can use the *Application Logs* to display and check if any errors occurred during runtime.

The *Application Logs* are a reuse library that is mainly used within the *Application Jobs* app, but can also be used in other apps. This reuse library gives you a clearly structured overview of all errors that might have occurred during runtime.

## Key Features

You can use this reuse library to:

- view application log details
- filter the logs by severity
- search for message texts
- display the message details

- display archived application logs

## Component for Customer Incidents

BC-SRV-APS-APL

For more information, see

- [Design Time API \[page 1537\]](#)
- [Runtime API \[page 1541\]](#)
- [How to Use the Fiori Reuse Library \[page 1602\]](#)

## Related Information

[Working with Application Log Objects](#)

### 4.2.11.9.1 Design Time API

You can use the *Application Log Design Time API* if you want to create, change, or delete an application log object or subobject during the development process of an application.

For information on how to create an application log object in ADT, see [Working with Application Log Objects](#).

The *Application Log Design Time API* provides the following design time operations:

- Create a new application log object, optionally with subobjects
- Delete an application log object and all of its subobjects
- Add a new subobject to an existing application log object
- Delete a subobject from an application log object
- Read an application log object and all of its subobjects

## Create a New Application Log Object, Optionally with Subobjects

Once you have created an instance of the `CL_BALI_OBJECT_HANDLER` API class using method `CREATE_OBJECT`, a new application log object is created.

### ↔ Sample Code

```
DATA(lo_log_object) = cl_bali_object_handler->get_instance( ).  
TRY.  
    lo_log_object->create_object( EXPORTING iv_object = 'MYOBJECT'  
                                    iv_object_text = 'My Application  
Log Object'
```

```

        iv_package = 'MYPACKAGE'
        iv_transport_request =
'MYTRANSPORT' ).
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.
```

In case the new application log object has subobjects, you can create these too, using the CREATE\_OBJECT method.

#### ↔ Sample Code

```

DATA: ls_subobject TYPE LINE OF if_bali_object_handler=>ty_tab_subobject,
      lt_subobject TYPE if_bali_object_handler=>ty_tab_subobject.
ls_subobject-subobject = 'MYSUBOBJECT'.
ls_subobject-subobject_text = 'My Application Log Sub-Object'.
APPEND ls_subobject TO lt_subobject.
DATA(lo_log_object) = cl_bali_object_handler=>get_instance( ).
TRY.
    lo_log_object->create_object( EXPORTING iv_object = 'MYOBJECT'
                                    iv_object_text = 'My Application
Log Object'
                                    it_subobjects = lt_subobject
                                    iv_package = 'MYPACKAGE'
                                    iv_transport_request =
'MYTRANSPORT' ).
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.
```

## Delete an Application Log Object and all of its Subobjects

Use method DELETE\_OBJECT to delete an application log object and all of its subobjects.

#### ↔ Sample Code

```

DATA(lo_log_object) = cl_bali_object_handler=>get_instance( ).
TRY.
    lo_log_object->delete_object( EXPORTING iv_object = 'MYOBJECT'
                                    iv_transport_request =
'MYTRANSPORT' ).
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.
```

## Add a New Subobject to an Existing Application Log Object

Use method ADD\_SUBOBJECT if you want to add a new subobject to an existing application log object.

#### ↔ Sample Code

```
DATA(lo_log_object) = cl_bali_object_handler=>get_instance( ).
```

```

TRY.
    lo_log_object->add_subobject( EXPORTING iv_object = 'MYOBJECT'
                                    iv_subobject = 'MYSUBOBJECT'
                                    iv_subobject_text = 'My
Application Log Sub-Object'
                                    iv_transport_request =
                                    'MYTRANSPORT' ).
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.

```

## Delete a Subobject from an Application Log Object

You can delete a subobject using method `DELETE_SUBOBJECT`.

### ↔ Sample Code

```

DATA(lo_log_object) = cl_bali_object_handler->get_instance( ).
TRY.
    lo_log_object->delete_subobject( EXPORTING iv_object = 'MYOBJECT'
                                    iv_subobject = 'MYSUBOBJECT'
                                    iv_transport_request =
                                    'MYTRANSPORT' ).
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.

```

## Read an Application Log Object and all of its Subobjects

Use method `READ_OBJECT` to read an application log object with its subobjects.

### ↔ Sample Code

```

DATA(lo_log_object) = cl_bali_object_handler->get_instance( ).
TRY.
    lo_log_object->read_object( EXPORTING iv_object = 'MYOBJECT'
                                IMPORTING et_subobjects = DATA(lt_subobjects)
                                ev_object_text =
                                DATA(lv_object_text) .
    CATCH cx_bali_objects INTO DATA(lx_exception).
        WRITE lx_exception->get_text( ).
ENDTRY.

```

## 4.2.11.9.1.1 Classes and Interfaces of the Design Time API

You can use the `CL_BALI_OBJECT_HANDLER` class as *Application Log Design Time API* to create, change, read, or delete application log objects or subobjects. It uses the public interface `IF_BALI_OBJECT_HANDLER`.

## Public Methods

GET\_INSTANCE (static)

Parameter	Type	Description
RO_OBJ_HANDLER	Returning	Create, change or delete application log objects

Create a new application log object, optionally with subobjects.

IF\_BALI\_OBJECT\_HANDLER~CREATE\_OBJECT

Parameter	Type	Description
IV_OBJECT	Importing	New application log object
IV_OBJECT_TEXT	Importing	Description for new application log object
IT_SUBOBJECTS	Importing	Table of new application log subobjects
IV_PACKAGE	Importing	Package
IV_TRANSPORT_REQUEST	Importing	Transport request

### ⓘ Note

The input of a table of subobjects for parameter `IT_SUBOBJECTS` is optional. If parameters `IV_PACKAGE` and `IV_TRANSPORT_REQUEST` are not provided, it is assumed that a local object shall be created.

Delete an application log object and all of its subobjects.

IF\_BALI\_OBJECT\_HANDLER~DELETE\_OBJECT

Parameter	Type	Description
IV_OBJECT	Importing	Application log object
IV_TRANSPORT_REQUEST	Importing	Transport request

Add a new subobject to an existing application log object.

IF\_BALI\_OBJECT\_HANDLER~ADD\_SUBOBJECT

Parameter	Type	Description
IV_OBJECT	Importing	Application log object
IV_SUBOBJECT	Importing	New application log subobject
IV_SUBOBJECT_TEXT	Importing	Description for new application log subobject

Parameter	Type	Description
IV_TRANSPORT_REQUEST	Importing	Transport request

Delete a subobject from an application log object.

IF\_BALI\_OBJECT\_HANDLER~DELETE\_SUBOBJECT

Parameter	Type	Description
IV_OBJECT	Importing	Application log object
IV_SUBOBJECT	Importing	Application log subobject
IV_TRANSPORT_REQUEST	Importing	Transport request

Read an application log object and all of its subobjects

IF\_BALI\_OBJECT\_HANDLER~READ\_OBJECT

Parameter	Type	Description
IV_OBJECT	Importing	Application log object
EV_OBJECT_TEXT	Importing	Description for an application log object
ET_SUBOBJECTS	Importing	Table of application log subobjects

#### ⓘ Note

For all methods, IV\_TRANSPORT\_REQUEST is not required for local objects.

All methods of IF\_BALI\_OBJECT\_HANDLER include exception CX\_BALI\_OBJECTS.

## 4.2.11.9.2 Runtime API

Use a class-based API to create and read application logs.

You can use a class-based API to create and read application logs. The application log is a tool for collecting messages or exceptions which are stored in a log. You can save this log in the database or read and delete it from there.

If an application wants to collect messages and exceptions which occur during runtime of an application program, the following steps are required:

- [Create a New Application Log \[page 1542\]](#)
- [Set Header Information \[page 1545\]](#)
- [Add Items \[page 1546\]](#)
- [Writing Application Logs to the Database \[page 1552\]](#)

If an application wants to read the content of an application log from the database, for example, to display its content, the following steps are required:

- [Define a Filter \[page 1554\]](#)
- [Read Application Logs from the Database \[page 1556\]](#)
- [Get Header Information from the Log \[page 1558\]](#)
- [Get Items from the Log \[page 1559\]](#)

## 4.2.11.9.2.1 Creating an Application Log

If an application wants to collect messages and exceptions which occur during runtime of an application program, the following steps are required:

- [Create a New Application Log \[page 1542\]](#)
- [Set Header Information \[page 1545\]](#)
- [Add Items \[page 1546\]](#)
- [Writing Application Logs to the Database \[page 1552\]](#)

### 4.2.11.9.2.1.1 Create a New Application Log

If you want to create a new application log, you must first create an object of class `CL_BALI_LOG`. This object manages a single application log and allows to get and set the log header and all log items, such as messages or exceptions.

To create an object of class `CL_BALI_LOG`, the class provides two methods:

- If the header information of the log, for example the external identifier, is already known when the application log is created, you can use the `CREATE_WITH_HEADER` method. In this case, you can set the log header during the call. For more information, see [Set Header Information \[page 1545\]](#).

#### ❖ Example

##### ❖ Sample Code

```
CLASS zcl_test_write DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_test_write IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    TRY.
      " Create a new application log
      DATA(l_log) = cl_bali_log=>create( ).
      " Add a header to the log
```

```

l_log->set_header( header = cl_bali_header_setter=>create( object =
'ZOBJECT'

subobject = 'ZSUBOBJECT'

external_id = 'External ID' ) .
    " Add a message as item to the log
    DATA(l_message) = cl_bali_message_setter=>create( severity =
if_bali_constants=>c_severity_error
                                                id = 'PO'
                                                number = '000' ).

l_log->add_item( item = l_message ).
    " Add a second message, this time from system fields SY-MSGID, ...
MESSAGE ID 'ZTEST' TYPE 'S' NUMBER '058' INTO DATA(l_text).
l_log->add_item( item = cl_bali_message_setter=>create_from_sy( ) ).
    " Add a free text to the log
    DATA(l_free_text) = cl_bali_free_text_setter=>create( severity =
if_bali_constants=>c_severity_error
                                                text = 'Some
Error Text' ).
    l_log->add_item( item = l_free_text ).
        " Add an exception to the log
        DATA: i TYPE i.
TRY.
    i = 1 / 0.
    CATCH cx_sy_zerodivide INTO DATA(l_ref).
ENDTRY.
DATA(l_exception) = cl_bali_exception_setter=>create( severity =
if_bali_constants=>c_severity_error
                                                exception =
l_ref ).
    l_log->add_item( item = l_exception ).
        " Save the log into the database
        cl_bali_log_db=>get_instance( )->save_log( log = l_log ).
        CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
            out->write( l_runtime_exception->get_text( ) ).
ENDTRY.
ENDMETHOD.
ENDCLASS.
```

## ↳ Sample Code

```

...
TRY.
    DATA(l_log) = cl_bali_log=>create_with_header(
                    header = cl_bali_header_setter=>create( object =
'ZOBJECT'
                                                subobject =
'ZSUBOBJECT'
                                                external_id =
'External ID' ) .
        CATCH cx_bali_runtime INTO DATA(l_exception).
            out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

If the header information is not known when the application log is created, you can use the `CREATE` method. It creates an empty application log. In this case, the header should be set later using the `SET_HEADER` method. If no subobjects were defined for the application log object, an empty string should be used at the parameter `SUBOBJECT`.

#### ↳ Sample Code

```
...
TRY.
    DATA(l_log) = cl_bali_log->create( ).
    ...
    l_log->set_header( header = cl_bali_header_setter->create( object =
'ZOBJECT'
                                         subobject
= 'ZSUBOBJECT'

external_id = 'External ID' ) .
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
...

```

If the application log is no longer needed in the memory, for example because it was saved into the database, it's possible to release its memory. This can be done using the method RELEASE\_MEMORY.

After calling the method RELEASE\_MEMORY, the log object is invalidated and can no longer be used. The method IS\_INVALIDATED can be used to check whether the memory of the log object was already released by a previous call of the method RELEASE\_MEMORY.

#### ↳ Sample Code

```
...
TRY.
    DATA(l_log) = cl_bali_log->create( ).

    ...
    " Add items to the log and, for example, write the log to the database
    ...
    " Release the memory of the log
    l_log->release_memory( ).

    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
...

```

#### ↳ Sample Code

```
...
TRY.
    DATA(l_log) = cl_bali_log->create( ).

    ...
    " Check whether the log was already invalidated
    IF l_log->is_invalidated( ) = abap_true.
        " Start some error handling
    ENDIF.

    ...
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
...

```

## 4.2.11.9.2.1.2 Set Header Information

An application log contains header information and attributes to identify the log. During log creation, the following header information can be set:

- Log Object (required): Identifies the application to which the log belongs.
- Log Subobject: An application can define subcategories of its log object. These subcategories are called subobjects and must be set if the application defined them. If no subobjects were defined for the log object, the field should remain empty.
- External Identifier: We recommend to enter your preferred identifier containing, for example, the application document number.
- Expiry Date: It defines the date when the log expires and can be deleted from the database. The default value is the creation date of the log + 7 days.
- A flag defining whether it should not be allowed to delete the log before the expiry date by a standard report. The application is still able to delete the log before this date.
- Context: Allows the storage of application-specific data related to the log. See [Set a Context \[page 1551\]](#).

To define the log header, an instance of the `IF_BALI_HEADER_SETTER` interface is required. To create this instance, you can use method `CREATE` of class `CL_BALI_HEADER_SETTER`. It allows to set the object, subobject and the external identifier of the application log.

Interface `IF_BALI_HEADER_SETTER` contains the following methods to set or change the header attributes:

- `SET_DESCRIPTOR`: Changes the object, subobject and external identifier.
- `SET_EXPIRY`: Sets the expiry date and the *keep until expiry* flag.
- `SET_CONTEXT`: Sets the context of the log.

### ① Note

You have two options to transfer the header to the application log object:

- Using method `CREATE_WITH_HEADER` of class `CL_BALI_LOG` when the log object is created.
- Using method `SET_HEADER` of interface `IF_BALI_LOG`.

If the header was already written to the log (for example, using method `SET_HEADER` of interface `IF_BALI_LOG`) and if the header object is changed afterwards (for example, using method `SET_DESCRIPTOR`), you must call method `SET_HEADER` to make these changes visible in the log.

### ↔ Sample Code

```
...
TRY.
    DATA(l_header) = cl_bali_header_setter->create( object = 'ZOBJECT'
                                                    subobject = 'ZSUBOBJECT'
                                                    external_id = 'External
ID' ).
    l_header->set_expiry( expiry_date = CONV
#( cl_abap_context_info->get_system_date( ) + 5 )
                           keep_until_expiry = abap_true ).
    CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
        out->write( l_runtime_exception->get_text( ) ).
ENDTRY.
...

```

### 4.2.11.9.2.1.3 Add Items

To write items, such as messages or exceptions, to the log during runtime, the application requires an item object which contains all attributes of the item, such as the message number or the reference to the exception object.

The following categories of log items are provided:

- Message (for example, a message is displayed using ABAP command **MESSAGE**). For more information, see [Create a Message \[page 1548\]](#).
- Free Text (a text with a maximum length of 200 characters). For more information, see [Create a Free Text \[page 1549\]](#).
- Exception (for example, an exception is raised using ABAP command **RAISE EXCEPTION** and can be caught using ABAP command **CATCH**). For more information, see [Create an Exception \[page 1550\]](#).

To add a log item to an application log, the **IF\_BALI\_LOG** interface provides the following methods:

- **ADD\_ITEM**: Append an item to the application log
- **CUMULATE\_ITEM**: If the item is a message, it's checked whether the log already contains a message with the same message attributes (severity, message ID, message number and variables). If yes, the counter of the message is increased by 1. In all other cases, it works the same way as **ADD\_ITEM**.
- **ADD\_MESSAGES\_FROM\_BAPIRETTAB**: Append several messages to the application log. The message attributes are stored in an internal table of type **BAPIRETTAB**.
- **ADD\_ABAP\_BEHAVIOR\_MESSAGE**: Append an ABAP behavior message, which uses the interface **IF\_ABAP\_BEHV\_MESSAGE**, to the log.
- **ADD\_ALL\_ITEMS\_FROM\_OTHER\_LOG**: Read all items from another application log and append them to this log. This method can be used if the items of different application logs are merged into one log. Sometimes it's required to filter items which are put into an application log by using the method **ADD\_ITEM**. For example, there may be an encapsulated application class which writes error and status messages into an application log, and the caller of this application class only requires the error messages. In this case, the caller can define an item filter (see [Define a Filter for the Log Items \[page 1551\]](#)). The caller can then add this item filter to the log using the method **SET\_FILTER\_FOR\_ADD\_ITEM**. If an item filter is set for a log object, only those items that pass this filter are added to the log object.

#### ⓘ Note

If an item was already added to the application log, its attributes can't be changed any more. Therefore, if the attributes of the item object are changed after the item was added to the log, these changes are not transferred to the log.

When an item is added to the application log, it can happen that it's converted internally. For example, an exception which is based on a message (which contains a message ID and message number) is internally converted into a message.

#### ↔ Sample Code

```
...
TRY.
  DATA(l_log) = cl_bali_log->create( ).

  MESSAGE ID 'ZTEST' TYPE 'W' NUMBER '002' INTO DATA(l_text).
  l_log->add_item( item = cl_bali_message_setter->create_from_sy( ) ).
```

```

l_log->cumulate_item( item = cl_bali_message_setter=>create_from_sy( ) ).  

DATA: i TYPE i.  

TRY.  

    i = 1 / 0.  

    CATCH cx_sy_zerodivide INTO DATA(l_ref).  

ENDTRY.  

l_log->add_item( item = cl_bali_exception_setter=>create( exception =  

l_ref ) ).  

    DATA(l_bapirettab) = VALUE bapirettab( ( id = 'BL' type = 'I' number =  

'315'                                         message_v1 = 'A' message_v2 =  

'B' message_v3 = 'C' message_v4 = 'D' )  

                                            ( id = 'BL' type = 'E' number =  

'319' ) ).  

    l_log->add_messages_from_bapirettab( message_table = l_bapirettab ).  

    CATCH cx_bali_runtime INTO DATA(l_exception).  

        out->write( l_exception->get_text( ) ).  

ENDTRY.  

...

```

#### ↳ Sample Code

```

...
TRY.  

    DATA(l_log_target) = cl_bali_log=>create( ).  

    MESSAGE ID 'ZTEST' TYPE 'W' NUMBER '002' INTO DATA(l_text).  

    l_log_target->add_item( item =  

cl_bali_message_setter=>create_from_sy( ) ).  

    DATA(l_log_source) = cl_bali_log=>create( ).  

    DATA: i TYPE i.  

TRY.  

    i = 1 / 0.  

    CATCH cx_sy_zerodivide INTO DATA(l_ref).  

ENDTRY.  

    l_log_source->add_item( item =  

cl_bali_exception_setter=>create( exception = l_ref ) ).  

    " add all items of l_log_source to l_log_target  

    l_log_target->add_all_items_from_other_log( source_log = l_log_source ).  

    CATCH cx_bali_runtime INTO DATA(l_exception).  

        out->write( l_exception->get_text( ) ).  

ENDTRY.  

...

```

#### ↳ Sample Code

```

...
TRY.  

    DATA(l_log) = cl_bali_log=>create( ).  

    " Add item filter to the log which only allows error messages  

    DATA(l_filter) = cl_bali_item_filter=>create( )-  

>set_severity( severity_table = VALUE #( ( 'E' ) ) ).  

    l_log->set_filter_for_add_item( filter = l_filter ).  

    MESSAGE ID 'ZTEST' TYPE 'W' NUMBER '002' INTO DATA(l_text).  

    l_log->add_item( item = cl_bali_message_setter=>create_from_sy( ) ).  "  

Message is not added, because it is a warning  

    MESSAGE ID 'ZTEST' TYPE 'E' NUMBER '002' INTO l_text.  

    l_log->add_item( item = cl_bali_message_setter=>create_from_sy( ) ).  "  

Message is added, because it is an error  

    " Get the current item filter  

    DATA(l_item_filter) = l_log->get_filter_for_add_item( ).  

    " Clear the current item filter

```

```

    l_log->set_filter_for_add_item( filter = value #( ) ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
    ...

```

### 4.2.11.9.2.1.3.1 Create a Message

One item category, which can be written into an application log, is a message. A message is identified by the message ID and the message number.

When an application log message is defined, you can set the following attributes:

- Severity, such as *Error*, *Warning*, *Information*. Possible values of the severity are defined as constants in interface `IF_BALI_CONSTANTS`.
- Message ID
- Message Number
- Variables 1 - 4 of the message
- Detail Level of the item: If the application displays the items of the application log, the detail level can be used to define at which level of detail the item shall be displayed.
- Context: Allows the storage of application-specific data related to the message. See [Set a Context \[page 1551\]](#).

To add a message to an application log, an instance of interface `IF_BALI_MESSAGE_SETTER` is required. To create this instance, class `CL_BALI_MESSAGE_SETTER` provides the following methods:

- `CREATE`: It allows to set the severity, message ID, message number and the variables of the message.
- `CREATE_FROM_SY`: You can use this method to create an application log message from the system fields in the list below. These system fields are filled, for example, by ABAP command `MESSAGE`. The following system fields are available:
  - SY-MSGTY
  - SY-MSGID
  - SY-MSGNO
  - SY-MSGV1
  - SY-MSGV2
  - SY-MSGV3
  - SY-MSGV4
- `CREATE_FROM_BAPIRET2`: It allows to set all message parameters via structure `BAPIRET2`.

Interface `IF_BALI_MESSAGE_SETTER` contains the following methods to set or change the attributes of the message:

- `SET_ATTRIBUTES`: Changes the severity, message ID, message number and the variables of the message.
- `SET_FROM_SY`: Changes the severity, message ID, message number and the variables of the message to the values of the system fields, such as `SY-MSGTY`.
- `SET_FROM_BAPIRET2`: Changes the severity, message ID, message number and the variables of the message to the values of structure `BAPIRET2`.

- SET\_DETAIL\_LEVEL: Sets the detail level.
- SET\_CONTEXT: Sets the context of the message.

#### ↔ Sample Code

```

...
    DATA(l_ref) = cl_bali_message_setter->create(
        severity = if_bali_constants->c_severity_error
        id = 'BL'
        number = '315'
        variable_1 = 'A'
        variable_2 = 'B'
        variable_3 = 'C'
        variable_4 = 'D' ).
    l_ref->set_detail_level( detail_level = '7' ).

...
MESSAGE ID 'ZTEST' TYPE 'I' NUMBER '315' WITH 'E' 'F' 'G' 'H' INTO
DATA(l_message).
    l_ref = cl_bali_message_setter->create_from_sy(
        )->set_detail_level( detail_level = '3' ).

...
DATA(l_bapiret2) = VALUE bapiret2( id = 'BL' type = 'I' number = '315'
    message_v1 = 'A' message_v2 = 'B'
    message_v3 = 'C' message_v4 = 'D' ).
    l_ref = cl_bali_message_setter->create_from_bapiret2( message_data =
l_bapiret2 ).
...

```

## 4.2.11.9.2.1.3.2 Create a Free Text

One item category, which can be written into an application log, is *Free Text*. It is any text with up to 200 characters.

The following attributes can be set when an application log free text is defined:

- Severity, such as *Error*, *Warning*, *Information*. Possible values of the severity are defined as constants in interface **IF\_BALI\_CONSTANTS**.
- Text content
- Detail level of the item: If the application displays the items of the application log, the detail level can be used to define at which level of detail the item should be visible.
- Context: Allows the storage of application-specific data related to the free text. See [Set a Context \[page 1551\]](#).

To add a free text to an application log, an instance of interface **IF\_BALI\_FREE\_TEXT\_SETTER** is required. To create this instance, you can use method **CREATE** of class **CL\_BALI\_FREE\_TEXT\_SETTER**. It allows to set the severity and the text content.

To set or change the attributes of the free text, interface **IF\_BALI\_FREE\_TEXT\_SETTER** contains the following methods:

- **SET\_TEXT**: Changes the text content and severity.
- **SET\_DETAIL\_LEVEL**: Sets the detail level.
- **SET\_CONTEXT**: Sets the context of the free text.

#### ↔ Sample Code

```
...
  DATA(l_ref) = cl_bali_free_text_setter->create(
    severity = if_bali_constants=>c_severity_error
    text = 'Some Error Text' ).
  l_ref->set_detail_level( detail_level = '4' ).
...
  l_ref = cl_bali_free_text_setter->create( text = 'Some Other Text'
    )->set_detail_level( detail_level = '1' ).
...
...
```

### 4.2.11.9.2.1.3.3 Create an Exception

One item category, which can be written into an application log, is an *Exception*. An exception is raised by the application using ABAP command **RAISE EXCEPTION** and can be caught by ABAP command **CATCH**.

To define an application log exception, you can set the following attributes:

- Severity, such as *Error*, *Warning*, *Information*. Possible values of the severity are defined as constants in interface **IF\_BALI\_CONSTANTS**.
- Reference to the exception object
- Detail Level of the item: If the application displays the items of the application log, the detail level can be used to define at which level of detail the item should be displayed.
- Context: Allows the storage of application-specific data related to the exception. See [Set a Context \[page 1551\]](#).

To add an exception to an application log, an instance of interface **IF\_BALI\_EXCEPTION\_SETTER** is required. To create this instance, method **CREATE** of class **CL\_BALI\_EXCEPTION\_SETTER** can be used. It allows to set the severity and the reference to the exception object.

If you want to set or change the attributes of the exception, you can use interface **IF\_BALI\_EXCEPTION\_SETTER**. It contains the following methods:

- **SET\_EXCEPTION**: Changes the reference to the exception object and the severity.
- **SET\_DETAIL\_LEVEL**: Sets the detail level.
- **SET\_CONTEXT**: Sets the context of the exception.

#### ↔ Sample Code

```
...
  DATA: i TYPE i.
  TRY.
    i = 1 / 0.
    CATCH cx_sy_zerodivide INTO DATA(l_exception_ref).
  ENDTRY.
  DATA(l_ref) = cl_bali_exception_setter->create( severity =
if_bali_constants=>c_severity_error
                                              exception =
l_exception_ref ).
  l_ref->set_detail_level( detail_level = '2' ).
...
...
```

## 4.2.11.9.2.1.3.4 Define a Filter for the Log Items

If items are put into an application log, there may be the requirement to use an item filter in order to, for example, keep the log small. For example, a report calls an encapsulated application class which writes different types of items into the application log. If the report doesn't need all items, but, for example, only the errors, it may define an item filter which only lets the errors pass.

To create a filter, an instance of interface `IF_BALI_ITEM_FILTER` is required. To create this instance, the method `CREATE` of class `CL_BALI_ITEM_FILTER` can be used. It creates an empty filter. The interface `IF_BALI_ITEM_FILTER` contains the following method to set the filter parameters:

- `SET_SEVERITY`: Set a filter for the severity (error, warning,...) of the item. The severity values which pass the filter are set via an internal table.

### ↔ Sample Code

#### Example:

Set a filter which lets all items pass that have the severity *Error* or *Warning*:

```
...
TRY.
  DATA(l_filter) = cl_bali_item_filter->create( ).
  l_filter->set_severity(severity_table = value #( ( 'E' )
( 'W' ) )).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

## 4.2.11.9.2.1.3.5 Set a Context

There may be the requirement to store application-specific data in the log. For example, a message may only be meaningful if you have information about the context in which a message was generated. An example of the context is the number of the order which was processed when the message was written into the log.

You can set a context for the complete log in the log header, and for each individual item of the log. To do this, the following steps are necessary:

- You need a structure which can store the context data. It must meet the following requirements:
  - The structure must either use a database table type, or a structure type which is defined in the data dictionary.
  - The structure type must be flat (which means that no internal tables or nested structures are allowed).
  - The structure type must be character-like (which means that no integer fields are allowed).
  - The total length of the complete structure must not be larger than 256 characters.
- Now, you can fill the structure fields with the application data which you want to store as context in the log or item.
- To transfer the structure with the context data to the log, you can use the method `SET_CONTEXT` of the header or item object.

## ↔ Sample Code

```
...
  DATA: l_context TYPE ADDRESS_1. " it can be any application structure that
  fulfils the requirements
  l_context = VALUE #( name1 = 'Test' name2 = 'Data' ). " fill the structure
  with the context data
  TRY.
    DATA(l_header) = cl_bali_header_setter->create( object = 'ZOBJECT'
                                                    subobject =
'ZSUBOBJECT' ).
    l_header->set_context( l_context ).
    CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
      out->write( l_runtime_exception->get_text( ) ).
  ENDTRY.
...
...
```

### 4.2.11.9.2.1.4 Writing Application Logs to the Database

You can use interface `IF_BALI_LOG_DB` to write an application log to the database or to delete database entries. To get an instance of the interface, method `GET_INSTANCE` of class `CL_BALI_LOG_DB` is available.

To change the logs in the database, interface `IF_BALI_LOG_DB` contains the following methods:

- `SAVE_LOG / SAVE_LOG_2ND_DB_CONNECTION`: Save an application log in the database. The log is identified by the log object (which uses interface `IF_BALI_LOG`).

It may be required to commit the saving of the log immediately after the saving. It ensures that the log is stored, even if the application calls `ROLLBACK WORK` later on. Whether it is required depends on the application. Some applications want to keep the log entries after the rollback, others want to remove everything by the `ROLLBACK WORK`, including the log entries.

Therefore, there are two different methods to save an application log:

- `SAVE_LOG` uses the default database connection. This means that the changes of the log are only committed if the application calls `COMMIT_WORK`.
- `SAVE_LOG_2ND_DB_CONNECTION` uses a service connection to the database for the saving, and it immediately commits the saving of the log.

In addition, the optional parameter `ASSIGN_TO_CURRENT_APPL_JOB` is available. If this parameter is set, and if the application runs in an application job, the connection between the application log and an application job is established. The application log is visible in the application job display.

- `DELETE_LOG`: Deletes an application log from the database. The log is identified by the log object (which uses interface `IF_BALI_LOG`).
- `ENQUEUE`: If there is a parallel processing of the same application report, it may happen - depending on the application - that several reports try to change the same application log at the same time. In this case, one report overwrites the application log data of the other report. To avoid this, method `ENQUEUE` can be used to set an enqueue on the application log. The log is identified by the log object (which uses interface `IF_BALI_LOG`).
- `DEQUEUE`: Clear the enqueue which was set via method `ENQUEUE`. The log is identified by the log object (which uses interface `IF_BALI_LOG`).

## ⓘ Note

You can quickly access a log stored in the database using the log handle. The log handle is the UUID of the log. To enable the application to store the log handle in one of the application tables, interface `IF_BAL_LOG` offers method `GET_HANDLE` which returns the log handle.

## ⚡ Example

Save a log in the database:

### ↗ Sample Code

```
...
TRY.
  DATA(l_log) =
    cl_bali_log->create_with_header( cl_bali_header_setter->create( object =
'ZOBJECT'
  subobject = 'ZSUBOBJECT'

  external_id = 'External ID' ) .
  MESSAGE ID 'ZTEST' TYPE 'I' NUMBER '315' WITH 'A' 'B' 'C' 'D' INTO
DATA(l_text).
  l_log->add_item( cl_bali_message_setter->create_from_sy( ) ).
  cl_bali_log_db->get_instance( )->save_log( log = l_log ).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

Save the same log using the second database connection:

### ↗ Sample Code

```
...
TRY.
  cl_bali_log_db->get_instance( )->save_log_2nd_db_connection( log =
l_log ).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

Delete the same log from the database:

### ↗ Sample Code

```
...
TRY.
  cl_bali_log_db->get_instance( )->delete_log( log = l_log ).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

Delete a log from the database which uses the handle `l_handle`.

## ↳ Sample Code

```
...  
    DATA l_handle TYPE if_bali_log=>ty_handle.  
    l_handle = ...  
    TRY.  
        DATA(l_log_db) = cl_bali_log_db->get_instance( ).  
        DATA(l_log) = l_log_db->load_log( handle = l_handle  
                                         read_only_header = abap_true ).  
        l_log_db->delete_log( log = l_log ).  
        CATCH cx_bali_runtime INTO DATA(l_exception).  
            out->write( l_exception->get_text( ) ).  
    ENDTRY.  
...
```

The following authorization is checked before a log is deleted from the database:

Authorization object: S\_APPL\_LOG, with

- ALG\_OBJECT: Object name of the application log
- ALG\_SUBOBJ: Subobject of the application log
- ACTVT: 06

## 4.2.11.9.2.2 Reading an Application Log

If an application wants to read the content of an application log from the database, for example, to display its content, the following steps are required:

- [Define a Filter \[page 1554\]](#)
- [Read Application Logs from the Database \[page 1556\]](#)
- [Get Header Information from the Log \[page 1558\]](#)
- [Get Items from the Log \[page 1559\]](#)

### 4.2.11.9.2.2.1 Define a Filter

If one or more application logs shall be read from the database and if the log handles are not known, you can identify the logs by defining a filter.

To create a filter, an instance of interface `IF_BALI_LOG_FILTER` is required. To create this instance, you can use method `CREATE` of class `CL_BALI_LOG_FILTER`. It creates an empty filter.

#### ⓘ Note

Due to performance reasons, we recommend to set at least the object and subobject in the filter.

Interface `IF_BALI_LOG_FILTER` contains the following methods to set the filter parameters:

- SET\_DESCRIPTOR: Set a filter for the log object, subobject and external identifier. It offers the following parameters:
  - Select a single object
  - Select a single subobject (wildcards allowed)
  - Select a table of subobjects (wildcards allowed)
  - Select a single external identifier (wildcards allowed)
  - Select a table of external identifiers (wildcards allowed)
- SET\_CREATE\_INFO: Set a filter for the attribute which identifies the creator of the log. It offers the following parameters:
  - Select a single user (wildcards allowed)
  - Select a table of users (wildcards allowed)
- SET\_TIME\_INTERVAL: Set a time interval for the creation date and time of the log. The start and end time of the interval are set using UTC time stamps.
- SET\_MAXIMUM\_LOG\_NUMBER: Set the maximum number of logs which shall be read from the database. If the parameter is not set, all logs are read (which fulfill the other filter criteria).

### ⌚ Example

Set a filter which selects up to 5 logs which were created in the last hour by the current user:

#### leftrightarrow Sample Code

```
...
TRY.
  DATA(l_filter) = cl_bali_log_filter->create( ).
  l_filter->set_create_info( user = sy-uname ).

  DATA(l_timestamp_now) = utclong_current( ).
  DATA(l_timestamp_minus_1_hour) = utclong_add( val = l_timestamp_now
                                              hours = '1-' ).
  l_filter->set_time_interval( start_time = l_timestamp_minus_1_hour
                               end_time = l_timestamp_now ).
  l_filter->set_maximum_log_number( max_log_number = 5 ).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

Set a filter which reads all logs of the current user with object ZOBJECT, subobject ZSUBOBJECT and an external identifier which starts with an 'E':

#### leftrightarrow Sample Code

```
...
TRY.
  DATA(l_filter) = cl_bali_log_filter->create(
                                              )->set_create_info( user = sy-
  uname
                                              )->set_descriptor( object =
  'ZOBJECT'
                                              subobject =
  'ZSUBOBJECT'
                                              external_id =
  'E*' ).
  CATCH cx_bali_runtime INTO DATA(l_exception).
    out->write( l_exception->get_text( ) ).
```

```
ENDTRY.  
...
```

## 4.2.11.9.2.2.2 Read Application Logs from the Database

You can use interface `IF_BALI_LOG_DB` to read one or more application logs from the database. To get an instance of the interface, method `GET_INSTANCE` of class `CL_BALI_LOG_DB` is available.

Interface `IF_BALI_LOG_DB` contains the following methods which can be used to read the logs from the database:

- `LOAD_LOG / LOAD_LOG_WITH_ITEMS`: Load one application log from the database.
  - The log is identified by the log handle.
  - The method returns an object of interface `IF_BALI_LOG`.
  - With method `LOAD_LOG_WITH_ITEMS`, the complete log including all items is read from the database. If you use method `LOAD_LOG` and set the optional parameter `READ_ONLY_HEADER`, only the log header is read from the database. The log items are not read yet.
- `LOAD_LOGS_VIA_FILTER / LOAD_LOGS_W_ITEMS_VIA_FILTER`: Load one or more logs from the database.
  - The logs are identified by a filter object of interface `IF_BALI_LOG_FILTER`.
  - The method returns an internal table of objects of interface `IF_BALI_LOG`.
  - With method `LOAD_LOG_W_ITEMS_VIA_FILTER`, the complete logs including all items are read from the database. If you use method `LOAD_LOGS_VIA_FILTER` and set the optional parameter `READ_ONLY_HEADER`, only the log headers are read from the database. The log items are not read yet.

The following authorization is checked before a log is read from the database:

Authorization object: `S_APPL_LOG`, with

- `ALG_OBJECT`: Object name of the application log
- `ALG_SUBOBJ`: Subobject of the application log
- `ACTVT`: 03

### Example

Load a log from the database:

### Sample Code

```
CLASS zcl_test_read DEFINITION  
  PUBLIC  
  FINAL  
  CREATE PUBLIC .  
  PUBLIC SECTION.  
    INTERFACES if_oo_adt_classrun.  
  PROTECTED SECTION.  
  PRIVATE SECTION.  
ENDCLASS.  
CLASS zcl_test_read IMPLEMENTATION.  
  METHOD if_oo_adt_classrun~main.  
    TRY.
```

```

    " Create a filter which selects all logs of the current user from
the last hour
    DATA(l_filter) = cl_bali_log_filter->create( ).
    l_filter->set_create_info( user = sy-uname ).
    DATA(l_timestamp_now) = utclong_current( ).
    DATA(l_timestamp_minus_1_hour) = utclong_add( val = l_timestamp_now
                                                hours = '1-' ).
    l_filter->set_time_interval( start_time = l_timestamp_minus_1_hour
                                end_time = l_timestamp_now ).
    " Read all Application Logs from the database which fit to the
filter
    DATA(log_table) = cl_bali_log_db->get_instance( )-
>load_logs_w_items_via_filter( filter = l_filter ).
    LOOP AT log_table INTO DATA(l_log).
        " Output log handle
        out->write( |Handle: { l_log->get_handle( ) }| ).
        " Get log header and output attributes of the header
        DATA(l_header) = l_log->get_header( ).
        out->write( |{ l_header->object } { l_header->subobject } |
{ l_header->external_id } { l_header->log_user }| ).
        " Get all items and output some data which exist in all item
categories
        DATA(l_item_table) = l_log->get_all_items( ).
        LOOP AT l_item_table INTO DATA(l_item_entry).
            out->write( |{ l_item_entry-log_item_number } { l_item_entry-
item->get_message_text( ) }| ).
            " Output attributes which are specific for messages and
exceptions
            IF l_item_entry-item->category =
if_bali_constants=>c_category_message.
                DATA(l_message_ref) = CAST
if_bali_message_getter( l_item_entry-item ).
                out->write( |{ l_message_ref->id } { l_message_ref-
number }| ).
            ELSEIF l_item_entry-item->category =
if_bali_constants=>c_category_exception.
                DATA(l_exception_ref) = CAST
if_bali_exception_getter( l_item_entry-item ).
                out->write( |{ l_exception_ref->exception_class } |
{ l_exception_ref->exception_id_name }| .
            ENDIF.
            ENDOOP.
        ENDDO.
        CATCH cx_bali_runtime INTO DATA(l_runtime_exception).
            out->write( l_runtime_exception->get_text( ) ).
        ENDTRY.
    ENDMETHOD.
ENDCLASS.

```

## Example

Load a single log from the database which uses log handle l\_handle. Only read the log header:

## Sample Code

```

...
    DATA l_handle TYPE if_bali_log=>ty_handle.
...
TRY.
    DATA(l_log) = cl_bali_log_db->get_instance( )->load_log( handle =
l_handle

read_only_header = abap_true ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).

```

```
ENDTRY.  
...
```

Load all logs from the database which were created by the current user:

#### ↔ Sample Code

```
...  
TRY.  
    DATA(l_filter) = cl_bali_log_filter->create( )-  
>set_create_info( user = sy-uname ).  
    DATA(log_table) = cl_bali_log_db->get_instance( )-  
>load_logs_w_items_via_filter( filter = l_filter ).  
    CATCH cx_bali_runtime INTO DATA(l_exception).  
        out->write( l_exception->get_text( ) ).  
    ENDTRY.  
...
```

### 4.2.11.9.2.2.3 Get Header Information from the Log

An application log contains header information and several attributes which identify the log. To read the header from a log, method `GET_HEADER` of interface `IF_BALI_LOG` can be used. It returns an object which uses interface `IF_BALI_HEADER_GETTER`.

Interface `IF_BALI_HEADER_GETTER` contains the following attributes:

- `OBJECT`: Log object which identifies the application to which the log belongs.
- `SUBOBJECT`: Log subobject which is the subcategory of an application log object.
- `EXTERNAL_ID`: A free text which is usually used by the application to identify the log. It contains, for example, the application document number.
- `LOG_TIMESTAMP`: UTC time stamp of the log (usually the creation time).
- `LOG_USER`: Log user (usually the user who created the log).
- `EXPIRY_DATE`: It defines the date when the log expires and can be deleted from the database.
- `KEEP_UNTIL_EXPIRY`: A flag which defines whether or not it is allowed to delete the log before the expiry date. If the flag is set, the standard reports which delete application logs do not delete the log before the expiry date. The application is still able to delete the log before this date.
- `CONTEXT_STRUCTURE_NAME`: The name of the dictionary structure which was used to set the context. See [Get the Context \[page 1563\]](#).
- `NUMBER_ALL_ITEMS`: Total number of items which are stored in the log.
- `NUMBER_ABORT_ITEMS`: Number of abort items which are stored in the log.
- `NUMBER_ERROR_ITEMS`: Number of error items which are stored in the log.
- `NUMBER_WARNING_ITEMS`: Number of warning items which are stored in the log.
- `NUMBER_INFORMATION_ITEMS`: Number of information items which are stored in the log.
- `NUMBER_STATUS_ITEMS`: Number of status items which are stored in the log.

Interface `IF_BALI_HEADER_GETTER` contains the following methods:

- `GET_OBJECT_DESCRIPTION`: Returns the description text of the log object in the logon language.

- GET\_SUBOBJECT\_DESCRIPTION: Returns the description text of the log subobject in the logon language.
- GET\_CONTEXT: Get the context of the log. See [Get the Context \[page 1563\]](#).

### Note

The header object contains the header attributes that were valid when the method GET\_HEADER was called. If the log is changed after this call (for example, when new items are added to the log), these changes are not automatically visible in the header object. To refresh the header object it's necessary to call the method GET\_HEADER again.

### Example

Load a single log from the database which uses log handle l\_handle, and output some fields of the log header:

#### Sample Code

```
...
DATA l_handle TYPE if_bali_log=>ty_handle.
...
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_header) = l_log->get_header( ).
    out->write( |{ l_header->object } { l_header-
>get_object_description( ) } {
        l_header->subobject } { l_header->external_id } {
        l_header->log_user }| ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
ENDTRY.
...
...
```

## 4.2.11.9.2.2.4 Get Items from the Log

After an application log was read from the database, the items stored in the log must be accessed. To read one or more items from an application log, interface **IF\_BALI\_LOG** provides the following methods:

- GET\_ITEM: Read a single item from the application log.
  - The item is identified by the *Log Item Number* which is a serial number containing the position of the item in the log.
  - The method returns the reference to the item object.
- GET\_ALL\_ITEMS: Read all items from the application log.
  - The method returns an internal table with the following structure:
    - The *Log Item Number* of the item
    - The reference to the item object

The item object contains all attributes of the item, for example the severity. Each item object uses interface **IF\_BALI\_ITEM\_GETTER**.

Public attributes of all log items:

- CATEGORY: Identifies the category of the item (see below).
- LOG\_ITEM\_NUMBER: The log item number which is the position of the item in the log.
- SEVERITY: Severity, such as 'Error', 'Warning', 'Information'. Possible values of the severity are defined as constants in interface `IF_BALI_CONSTANTS`.
- DETAIL\_LEVEL: If the application outputs the items of the application log, the detail level can be used to define at which level of detail the item shall be visible.
- TIMESTAMP: A UTC timestamp which contains the creation time of the item.
- CONTEXT\_STRUCTURE\_NAME: The name of the dictionary structure which was used to set the context. See [Get the Context \[page 1563\]](#).

Methods of all log items:

- GET\_MESSAGE\_TEXT: It returns the message text of the item. It is:
  - The output of ABAP command `MESSAGE` for a message item
  - The text of a free text item
  - The output of method `GET_TEXT` of the exception object for an exception item
- GET\_CONTEXT: Get the context of the item. See [Get the Context \[page 1563\]](#).

The category of the item restricts which other attributes of the item are available. For example, the message number is only available for a message, because a free text does not have a message number. To get additional attributes, a down cast to the more specific interface of the item category is required. The following item categories are supported:

- Message
  - Attribute `CATEGORY` contains the value `IF_BALI_CONSTANTS=>C_CATEGORY_MESSAGE`.
  - Interface of a message is `IF_BALI_MESSAGE_GETTER`.
- For more information, see [Get a Message \[page 1562\]](#).
- Free Text
  - Attribute `CATEGORY` contains the value `IF_BALI_CONSTANTS=>C_CATEGORY_FREE_TEXT`.
  - Interface of a free text is `IF_BALI_FREE_TEXT_GETTER`.
- For more information, see [Get a Free Text \[page 1561\]](#).
- Exception
  - Attribute `CATEGORY` contains the value `IF_BALI_CONSTANTS=>C_CATEGORY_EXCEPTION`.
  - Interface of an exception is `IF_BALI_EXCEPTION_GETTER`.
- For more information, see [Get an Exception \[page 1562\]](#).

## ⌚ Example

Load a single log from the database which uses log handle `l_handle` and output all items of the log:

### ⌚ Sample Code

```
...
DATA l_handle TYPE if_bali_log=>ty_handle.
...
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_item_table) = l_log->get_all_items( ).
    LOOP AT l_item_table INTO DATA(l_item_entry).
```

```

        out->write( |{ l_item_entry-log_item_number } { l_item_entry-item-
>get_message_text( ) }| ).
        IF l_item_entry-item->category =
if_bali_constants=>c_category_message.
            DATA(l_message_ref) = CAST if_bali_message_getter( l_item_entry-
item ). 
            out->write( |{ l_message_ref->id } { l_message_ref->number }| ).
        ELSEIF l_item_entry-item->category =
if_bali_constants=>c_category_exception.
            DATA(l_exception_ref) = CAST
if_bali_exception_getter( l_item_entry-item ). 
            out->write( |{ l_exception_ref->exception_class }
{ l_exception_ref->exception_id_name }| ).
        ENDIF.
    ENDLOOP.
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
    ...

```

#### 4.2.11.9.2.2.4.1 Get a Free Text

Similar to all other items, a free text can be read from the application log using methods `GET_ITEM` and `GET_ALL_ITEMS` of interface `IF_BALI_LOG`. The free text object, which is returned by these methods, uses interface `IF_BALI_FREE_TEXT_GETTER`.

The interface supports all attributes and methods of interface `IF_BALI_ITEM_GETTER`.

Interface `IF_BALI_FREE_TEXT_GETTER` contains no additional attributes.

##### ❖ Example

Load a single log from the database which uses log handle `l_handle` and output the text of the first item:

##### ❖ Sample Code

```

...
DATA l_handle TYPE if_bali_log=>ty_handle.
...
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_item) = l_log->get_item( log_item_number = '1' ).
    out->write( |{ l_item->get_message_text( ) }| ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
...

```

## 4.2.11.9.2.2.4.2 Get a Message

Similar to all other items, a message can be read from the application log using methods `GET_ITEM` and `GET_ALL_ITEMS` of interface `IF_BALI_LOG`. The message object, which is returned by these methods, uses interface `IF_BALI_MESSAGE_GETTER`.

The interface supports all attributes and methods of interface `IF_BALI_ITEM_GETTER`.

Interface `IF_BALI_MESSAGE_GETTER` contains the following additional attributes:

- ID: Message ID
- NUMBER: Message Number
- VARIABLE\_1: Variable 1 of the message
- VARIABLE\_2: Variable 2 of the message
- VARIABLE\_3: Variable 3 of the message
- VARIABLE\_4: Variable 4 of the message
- COUNT: The count of the message. It is increased, if method `CUMULATE_ITEM` of interface `IF_BALI_LOG` is used to add a message to the log and if the log already contains a message with the same message attributes (severity, message ID, message number and variables).

### Example

Load a single log from the database which uses log handle `l_handle` and output some attributes of the first item, if it is a message:

### Sample Code

```
...
DATA l_handle TYPE if_bali_log=>ty_handle.
...
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_item) = l_log->get_item( log_item_number = '1' ).
    IF l_item->category = if_bali_constants=>c_category_message.
        DATA(l_message_ref) = CAST if_bali_message_getter( l_item ).
        out->write( |{ l_message_ref->id } { l_message_ref->number }| ).
    ENDIF.
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

## 4.2.11.9.2.2.4.3 Get an Exception

Similar to all other items, an exception can be read from the application log using methods `GET_ITEM` and `GET_ALL_ITEMS` of interface `IF_BALI_LOG`. The exception object, which is returned by these methods, uses interface `IF_BALI_EXCEPTION_GETTER`.

The interface supports all attributes and methods of interface `IF_BALI_ITEM_GETTER`.

Interface `IF_BALI_EXCEPTION_GETTER` contains the following additional attributes:

- `EXCEPTION_CLASS`: The name of the exception class.
- `EXCEPTION_ID_NAME`: Name of the text ID of the ABAP exception.

#### ❖ Example

Load a single log from the database which uses log handle `l_handle` and output some attributes of the first item, if it is an exception:

#### ❖ Sample Code

```
...
DATA l_handle TYPE if_bali_log=>ty_handle.
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_item) = l_log->get_item( log_item_number = '1' ).
    IF l_item->category = if_bali_constants=>c_category_exception.
        DATA(l_exception_ref) = CAST if_bali_exception_getter( l_item ).
        out->write( |{ l_exception_ref->exception_class } { l_exception_ref-
>exception_id_name }| ).
    ENDIF.
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
    ENDTRY.
...
...
```

## 4.2.11.9.2.2.4.4 Get the Context

To read the context data from the log header or from the log item, the following steps are necessary:

- You must define a structure which can be used to store the context data. It's recommended that you use the same structure type that was used when the context was set.
  - To get the data dictionary name of the structure type that was used to set the context, you can use the attribute `CONTEXT_STRUCTURE_NAME` of the header or item object.
- After this, you can use the method `GET_CONTEXT` of the header or item object to fill this context structure.
  - If you use a different structure type for getting the context than the one used for setting, only the identically named components are copied to each other.

#### ❖ Sample Code

```
...
DATA: l_handle TYPE if_bali_log=>ty_handle.
DATA: l_context TYPE ADDRESS_1. " the structure type which was used to set
the context
...
TRY.
    DATA(l_log) = cl_bali_log_db=>get_instance( )->load_log( handle =
l_handle ).
    DATA(l_header) = l_log->get_header( ).
    l_header->get_context( IMPORTING context = l_context ).
    out->write( |{ l_context-name1 } { l_context-name2 }| ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
```

```

        out->write( l_exception->get_text( ) ) .
ENDTRY.
...

```

### 4.2.11.9.2.3 Classes and Interfaces of the Application Log API

The following classes and interfaces are available:

Access the Database

Class Name	Public Interface	Description
CL_BALI_LOG_DB	IF_BALI_LOG_DB	Handles database access like reading or writing of logs in the database.
CL_BALI_LOG_FILTER	IF_BALI_LOG_FILTER	Defines a filter for reading of logs from the database.

Access the Content of a Log

Class Name	Public Interface	Description
CL_BALI_LOG	IF_BALI_LOG	Reads and writes the header and items of a log

Writing the Log Header

Class Name	Public Interface	Description
CL_BALI_HEADER_SETTER	IF_BALI_HEADER_SETTER	Log header which can be put into a log

Reading the Log Header

Class Name	Public Interface	Description
	IF_BALI_HEADER_GETTER	Log header which was read from the log

Writing a Log Item

Class Name	Public Interface	Description
	IF_BALI_ITEM_SETTER	Each item contains this interface
CL_BALI_MESSAGE_SETTER	IF_BALI_MESSAGE_SETTER	Message which can be put into a log
CL_BALI_FREE_TEXT_SETTER	IF_BALI_FREE_TEXT_SETTER	Free text which can be put into a log
CL_BALI_EXCEPTION_SETTER	IF_BALI_EXCEPTION_SETTER	Exception which can be put into a log

## Reading a Log Item

Class Name	Public Interface	Description
	IF_BALI_ITEM_GETTER	Each item contains this interface
	IF_BALI_MESSAGE_GETTER	Message which was read from the log
	IF_BALI_FREE_TEXT_GETTER	Free text which was read from the log
	IF_BALI_EXCEPTION_GETTER	Exception which was read from the log

## Other Classes and Interfaces

Class Name	Public Interface	Description
	IF_BALI_CONSTANTS	Some constants, such as available item categories and severities
CL_BALI_ITEM_FILTER	IF_BALI_ITEM_FILTER	Define an item filter for adding items to a log

If one of the class methods can't be processed or can't return the requested results, an exception is raised. The following exceptions are possible, each of them inherit from exception class CX\_BALI\_RUNTIME:

### Exception Classes

Class Name	Description
CX_BALI_INVALID_PARAMETER	An input parameter of the method is invalid (e.g. the log object doesn't exist).
CX_BALI_NOT_FOUND	The entry which shall be read or changed was not found.
CX_BALI_NOT_POSSIBLE	<p>The requested processing is not possible.</p> <p>Possible values of class attribute ERROR_CODE:</p> <ul style="list-style-type: none"> <li>• CX_BALI_NOT_POSSIBLE=&gt;NO_AUTHORIZATION: No authorization to access the log</li> <li>• CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• CX_BALI_NOT_POSSIBLE=&gt;TOO_MANY_ITEMS: The maximum number 999999 of items was reached</li> <li>• CX_BALI_NOT_POSSIBLE=&gt;SAVE_NOT_ALLOWED: Error during saving to the database (e.g. database error or object is empty)</li> <li>• CX_BALI_NOT_POSSIBLE=&gt;ENTRY_IS_LOCKED: The enqueue cannot be set, because the log is already locked</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

### Note

Find more information about classes and interfaces of the [Application Log API](#) in the ABAP development tools for Eclipse (ADT).

## 4.2.11.9.2.3.1 CL\_BALI\_LOG\_DB (Interface IF\_BALI\_LOG\_DB)

Class CL\_BALI\_LOG\_DB handles all database accesses of the application logs. This includes the reading of one or several logs from the database, writing a log to the database and deleting a log from the database. In addition, it offers methods to set and clear an SAP enqueue on a log. The public interface of the instance methods is IF\_BALI\_LOG\_DB.

### Public Methods

Get an Instance of the Database Handler:

GET\_INSTANCE (static)

Name	Description
<b>Returning parameter</b>	
DB_HANDLER	Database handler object: A reference to interface IF_BALI_LOG_DB

Load a single log from the database into the memory:

LOAD\_LOG

Name	Description
<b>Importing parameters</b>	
HANDLE	Handle of the Application Log which shall be read
READ_ONLY_HEADER	(Optional): If set, only the header of the log is read (no items) Default: Not set
<b>Returning parameter</b>	
LOG	Log which was read from the database: Reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	<ul style="list-style-type: none"><li>The log handle is initial</li><li>The log was not found in the database</li></ul>

Name	Description
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;NO_AUTHORIZATION: No authorization to access the log</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

It is recommended to use method `LOAD_LOG_WITH_ITEMS` if all items should be loaded.

The following authorization is checked:

Authorization Object: S\_APPL\_LOG with:

- ALG\_OBJECT: Object name of the application log
- ALG\_SUBOBJ: Subobject of the application log
- ACTVT: 03

Load a single log including all items from the database into the memory:

`LOAD_LOG_WITH_ITEMS`

Name	Description
<b>Importing parameters</b>	
HANDLE	Handle of the application log that should be read
<b>Returning parameter</b>	
LOG	Log that was read from the database: Reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	<ul style="list-style-type: none"> <li>• The log handle is initial</li> <li>• The log was not found in the database</li> </ul>
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;NO_AUTHORIZATION: No authorization to access the log</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

## ⓘ Note

The following authorization is checked:

Authorization Object: S\_APPL\_LOG with:

- ALG\_OBJECT: Object name of the application log
- ALG\_SUBOBJ: Subobject of the application log
- ACTVT: 03

Load several logs via a filter from the database into the memory:

LOAD\_LOGS\_VIA\_FILTER

Name	Description
<b>Importing parameters</b>	
FILTER	Log filter object: Reference to interface IF_BALI_LOG_FILTER
READ_ONLY_HEADER	(Optional): If set, only the headers of the logs are read (no items) Default: Not set
<b>Returning parameter</b>	
LOG_TABLE	Table of logs which were read from the database: Table of references to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	No log can be returned which fits to the filter criteria
CX_BALI_INVALID_PARAMETERS	<ul style="list-style-type: none"><li>• The filter contains invalid values</li><li>• The filter is initial or empty</li></ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

## ⓘ Note

It is recommended to use method LOAD\_LOGS\_W\_ITEMS\_VIA\_FILTER if all items should be loaded.

The following authorization is checked:

Authorization Object: S\_APPL\_LOG with:

- ALG\_OBJECT: Object name of the application log
- ALG\_SUBOBJ: Subobject of the application log
- ACTVT: 03

If the object of a log is not allowed or if the log doesn't pass the authorization check, it is removed from the table of logs which is returned. An exception is only raised if the final table is empty.

Load several logs including all items via a filter from the database into the memory:

LOAD\_LOGS\_W\_ITEMS\_VIA\_FILTER

Name	Description
<b>Importing parameters</b>	
FILTER	Log filter object: Reference to interface IF_BALI_LOG_FILTER
<b>Returning parameter</b>	
LOG_TABLE	Table of logs which were read from the database: Table of references to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	No log can be returned that fits to the filter criteria
CX_BALI_INVALID_PARAMETERS	<ul style="list-style-type: none"><li>The filter contains invalid values</li><li>The filter is initial or empty</li></ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

The following authorization is checked:

Authorization Object: S\_APPL\_LOG with:

- ALG\_OBJECT: Object name of the application log
- ALG\_SUBOBJ: Subobject of the application log
- ACTVT: 03

If the object of a log is not allowed or if the log doesn't pass the authorization check, it's removed from the table of logs that is returned. An exception is only raised if the final table is empty.

Save a single log to the database using the default database connection:

SAVE\_LOG

Name	Description
<b>Importing parameters</b>	
LOG	Log that should be saved: Reference to interface IF_BALI_LOG
<b>Deprecated: Please use method SAVE_LOG_2ND_DB_CONNECTION instead</b>	
ASSIGN_TO_CURRENT_APPL_JOB	(Optional): If set, and if the application runs in an application job, a connection between the application log and an application job is established
	Default: Not set
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	

Name	Description
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;SAVE_NOT_ALLOWED: <ul style="list-style-type: none"> <li>• Log object and log subobject are empty</li> <li>• Locking of the log is not possible</li> <li>• Error during saving into the database</li> </ul> </li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

If parameter LOG is initial, the method returns without exception.

Save a single log to the database using a service connection to the database:

SAVE\_LOG\_2ND\_DB\_CONNECTION

Name	Description
<b>Importing parameters</b>	
LOG	Log that should be saved: Reference to interface IF_BALI_LOG
ASSIGN_TO_CURRENT_APPL_JOB	(Optional): If set, and if the application runs in an application job, a connection between the application log and an application job is established  Default: Not set
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	

Name	Description
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;SAVE_NOT_ALLOWED: <ul style="list-style-type: none"> <li>• Log object and log subobject are empty</li> <li>• Locking of the log is not possible</li> <li>• Error during saving into the database</li> </ul> </li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

A commit is executed on the service connection after the saving.

If parameter LOG is initial, the method returns without exception.

Delete a single log from the database:

DELETE\_LOG

Name	Description
<b>Importing parameters</b>	
LOG	Log which shall be saved: Reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;NO_AUTHORIZATION: No authorization to access the log</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_NOT_FOUND	The log was not found in the database
CX_BALI_INTERNAL_ERROR	Internal error during processing

## ⓘ Note

- If parameter LOG is initial, the method returns without exception.
- The following authorization is checked:  
Authorization Object: S\_APPL\_LOG with:
  - ALG\_OBJECT: Object name of the application log
  - ALG\_SUBOBJ: Subobject of the application log
  - ACTVT: 06

Set an SAP enqueue on a log:

ENQUEUE

Name	Description
<b>Importing parameters</b>	
LOG	Log which shall be saved: Reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"><li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li><li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;ENTRY_IS_LOCKED: The enqueue can't be set, because the log is already locked</li><li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li></ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

## ⓘ Note

If parameter LOG is initial, the method returns without exception.

Clear an SAP enqueue from a log:

DEQUEUE

Name	Description
<b>Importing parameters</b>	
LOG	Log which shall be saved: Reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	

Name	Description
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

#### ⓘ Note

If parameter LOG is initial, the method returns without exception.

## 4.2.11.9.2.3.2 CL\_BALI\_LOG\_FILTER (Interface IF\_BALI\_LOG\_FILTER)

Class CL\_BALI\_LOG\_FILTER allows to define a filter which can be used if logs shall be read from the database and if the log handles are not known. The public interface of the instance methods is IF\_BALI\_LOG\_FILTER.

### Public Methods

Create an instance of the filter class:

CREATE (static)

Name	Description
<b>Returning parameter</b>	
FILTER	Filter object: A reference to interface IF_BALI_LOG_FILTER

Set object, subobject and external identifier of the log. It overwrites all previous filter settings of object, subobject and external identifier:

SET\_DESCRIPTOR

Name	Description
<b>Importing parameters</b>	
OBJECT	(Optional): Object of the log (no wildcards)
SUBOBJECT	(Optional): Subobject of the log (wildcards allowed)
SUBOBJECT_TABLE	(Optional): Table with subobjects (wildcards allowed)

Name	Description
EXTERNAL_ID	(Optional): External identifier of the log (wildcards allowed)
EXTERNAL_ID_TABLE	(Optional): Table with external identifiers (wildcards allowed)
<b>Returning parameter</b>	
NEW_FILTER	Reference to current filter object
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<p>ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED:</p> <p>Access to the log object is not allowed</p>

#### ⓘ Note

If parameter EXTERNAL\_ID is supplied and empty, the filter searches for logs with empty external identifier.  
If parameters OBJECT, SUBOBJECT, SUBOBJECT\_TABLE, or EXTERNAL\_ID\_TABLE are supplied and empty, they are ignored

Set information about the log creation like the user. It overwrites all previous filter settings about the log creation:

SET\_CREATE\_INFO

Name	Description
<b>Importing parameters</b>	
USER	(Optional): Log user (wildcards allowed)
USER_TABLE	(Optional): Table with log users (wildcards allowed)
<b>Returning parameter</b>	
NEW_FILTER	Reference to current filter object

#### ⓘ Note

If parameters USER or USER\_TABLE are supplied and empty, they are ignored.

Set the date and time interval of the log creation. It overwrites all previous filter settings of the time interval:

SET\_TIME\_INTERVAL

Name	Description
<b>Importing parameters</b>	
START_TIME	UTC time stamp of the start time of the time interval

Name	Description
END_TIME	UTC time stamp of the end time of the time interval
<b>Returning parameter</b>	
NEW_FILTER	Reference to current filter object
Set the maximum number of logs which are processed:	
SET_MAXIMUM_LOG_NUMBER	
Name	Description
<b>Importing parameters</b>	
MAX_LOG_NUMBER	Maximum number of logs which are processed (0 = all logs are processed which is the default)
<b>Returning parameter</b>	
NEW_FILTER	Reference to current filter object
Get all filter values:	
GET_ALL_VALUES	
Name	Description
<b>Exporting parameters</b>	
OBJECT_TABLE	Table of objects (no wildcards)
SUBOBJECT_TABLE	Table of subobjects (wildcards allowed)
EXTERNAL_ID_TABLE	Table of external identifiers (wildcards allowed)
USER_TABLE	
TIME_INTERVAL	Date and time interval
MAX_LOG_NUMBER	Maximum number of logs processed

## 4.2.11.9.2.3.3 CL\_BALI\_LOG (Interface IF\_BALI\_LOG)

Class `CL_BALI_LOG` handles all read and change operations on a single application log. It contains methods to read and change the log header. In addition, it allows to read items from the log and to add items to the log. The public interface of the instance methods is `IF_BALI_LOG`.

### Public Methods

Create an instance of the log class:

CREATE (static)

Name	Description
<b>Returning parameter</b>	
LOG	Log object: A reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_INTERNAL_ERROR	Internal error during processing

Create an instance of the log class and set the header:

CREATE\_WITH\_HEADER (static)

Name	Description
<b>Importing parameter</b>	
HEADER	Header which is put into the log: Reference to interface IF_BALI_HEADER_SETTER
<b>Returning parameter</b>	
LOG	Log object: A reference to interface IF_BALI_LOG
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>OBJECT_NOT_ALLOWED: Access to the log object of the header is not allowed
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header don't exist
CX_BALI_INTERNAL_ERROR	Internal error during processing

Get the log handle which is the unique identifier of the log:

GET\_HANDLE

Name	Description
<b>Returning parameter</b>	
HANDLE	Log handle

Get the log header:

GET\_HEADER

Name	Description
<b>Returning parameter</b>	
HEADER	Log header: References to interface IF_BALI_HEADER_GETTER

Name	Description
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<p>ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED:</p> <p>The memory of the log was released. The log can't be used</p>
CX_BALI_INTERNAL_ERROR	Internal error during processing

Set the log header:

SET\_HEADER

Name	Description
<b>Importing parameter</b>	
HEADER	Header which is put into the log: References to interface IF_BALI_HEADER_SETTER
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object of the header is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header don't exist
CX_BALI_INTERNAL_ERROR	Internal error during processing

Add an item (e.g. a message) to the log:

ADD\_ITEM

Name	Description
<b>Importing parameter</b>	
ITEM	Item which is added: Reference to interface IF_BALI_ITEM_SETTER
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	

Name	Description
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object of the header is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header don't exist
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

An exception may contain a message. This means that the ABAP exception class contains interface `IF_T100_MESSAGE`, or it contains at least the attributes `T100_MSGID` and `T100_MSGNO`. In this case, the exception is internally converted into a message before it is added to the log.

If the item is a message, it's checked whether the log already contains another message with identical message attributes. These are the attributes: severity, message ID, message number and message variable 1 - 4. If this message exist, the message counter of the message is increased by 1. Otherwise, a new message is added to the log. Also free texts and exceptions are always added to the log without cumulation.

### CUMULATE\_ITEM

Name	Description
<b>Importing parameter</b>	
ITEM	Item which is cumulated or added: Reference to interface <code>IF_BALI_ITEM_SETTER</code>
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED: Access to the log object of the header is not allowed</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header don't exist
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

An exception may contain a message. This means that the ABAP exception class contains interface `IF_T100_MESSAGE`, or it contains at least the attributes `T100_MSGID` and `T100_MSGNO`. In this case,

the exception is internally converted to a message before it is added to the log (but this message is not cumulated to an already existing message).

Add all messages from an internal table of type BAPIRETTAB table to the log:

ADD\_MESSAGES\_FROM\_BAPIRETTAB

Name	Description
<b>Importing parameter</b>	
MESSAGE_TABLE	An internal table with messages which use type BAPIRETTAB
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSI-BLE=&gt;TOO_MANY_ITEMS: The maximum number 999999 of items was reached</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSI-BLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The message ID of one of the messages is initial
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

If a message of the message table can't be added to the log, it's skipped and the processing continues until the end of the table is reached. Afterwards, an exception is raised to notify the caller that some of the messages couldn't be added to the log.

Add a message from the ABAP Restful Application Programming Model (interface IF\_ABAP\_BEHV\_MESSAGE) to the log:

ADD\_ABAP\_BEHAVIOR\_MESSAGE

Name	Description
<b>Importing parameter</b>	
MESSAGE	A reference to the interface IF_ABAP_BEHV_MESSAGE
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSI-BLE=&gt;TOO_MANY_ITEMS: The maximum number 999999 of items was reached</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSI-BLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The message ID of the messages is initial

Name	Description
CX_BALI_INTERNAL_ERROR	Internal error during processing
Add all items from another log to this log:	
ADD_ALL_ITEMS_FROM_OTHER_LOG	
Name	Description
<b>Importing parameter</b>	
SOURCE_LOG	Reference to the log whose items are to be copied
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	<ul style="list-style-type: none"> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;TOO_MANY_ITEMS: The maximum number of 999999 items was reached</li> <li>• ERROR_CODE: CX_BALI_NOT_POSSIBLE=&gt;LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used</li> </ul>
CX_BALI_INVALID_PARAMETER	The source log is invalid. It contains a log handle which doesn't exist
CX_BALI_INTERNAL_ERROR	Internal error during processing

### ⓘ Note

If the parameter SOURCE\_LOG is initial, the processing is skipped and no exception is raised.

Add an item filter to the log. The item filter is checked before an item is added to the log using the methods ADD\_ITEM, CUMULATE\_ITEM, ADD\_MESSAGES\_FROM\_BAPIRETTAB, or ADD\_ABAP\_BEHAVIOR\_MESSAGE. If an item doesn't pass the filter, it's ignored.

SET\_FILTER\_FOR\_ADD\_ITEM

Name	Description
<b>Importing parameter</b>	
FILTER	A reference to the item filter
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used

### ⓘ Note

- If FILTER is initial, any item filter is removed from the log. The log then no longer uses an item filter.
- Since the method ADD\_ALL\_ITEMS\_FROM\_OTHER\_LOG is intended for the mass-processing of items, it ignores the item filter because of performance reasons.

- The filter is only used by methods of the class CL\_BALI\_LOG. If an application uses the application log function modules to add an item to the log, the filter is ignored.

Get the item filter which was set using SET\_FILTER\_FOR\_ADD\_ITEM.

#### GET\_FILTER\_FOR\_ADD\_ITEM

Name	Description
<b>Returning parameter</b>	
FILTER	A reference to the item filter which is currently used by the log
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used

Get a single item from the log:

#### GET\_ITEM

Name	Description
<b>Importing parameter</b>	
LOG_ITEM_NUMBER	Serial number of the item which shall be read (it is the position of the item in the log)
<b>Returning parameter</b>	
ITEM	Item which was read: Reference to interface IF_BALI_ITEM_GETTER
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	An item with the requested log item number does not exist
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used
CX_BALI_INTERNAL_ERROR	Internal error during processing

Get all items from the log:

#### GET\_ALL\_ITEMS

Name	Description
<b>Returning parameter</b>	

Name	Description
ITEM_TABLE	<p>Table of all items which are stored in the log. The table has the following structure:</p> <ul style="list-style-type: none"> <li>• LOG_ITEM_NUMBER: The serial number of the item in the log</li> <li>• ITEM: Item object: Reference to interface IF_BALI_ITEM_GETTER</li> </ul>
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_FOUND	No item was found in the log
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>LOG_WAS_INVALIDATED: The memory of the log was released. The log can't be used
CX_BALI_INTERNAL_ERROR	Internal error during processing

#### **RELEASE\_MEMORY:**

Remove the log from the memory. After this, the log is invalidated and can no longer be used.

Check whether the memory of the log was released in order to verify that the log is invalidated and can no longer be used.

IS\_INVALIDATED

Name	Description
<b>Returning parameter</b>	
IS_INVALIDATED	If set, the log is invalidated and can no longer be used

### **4.2.11.9.2.3.4 CL\_BALI\_HEADER\_SETTER (Interface IF\_BALI\_HEADER\_SETTER)**

Class CL\_BALI\_HEADER\_SETTER is used to set the header attributes of an application log, such as the log object, subobject, and the external identifier. The public interface of the instance methods is IF\_BALI\_HEADER\_SETTER.

#### **Public Methods**

Create an instance of the header class with the settings of the descriptor (object, subobject and external identifier):

CREATE (static)

Name	Description
<b>Importing parameters</b>	
OBJECT	Object of the log
SUBOBJECT	Subobject of the log If no subobjects were defined for the log object, the parameter should be empty
EXTERNAL_ID	(Optional): External identifier of the log Default: ''
<b>Returning parameter</b>	
HEADER	Header object: A reference to interface IF_BALI_HEADER_SETTER
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>OBJECT_NOT_ALLOWED: Access to the log object is not allowed
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header doesn't exist

Set object, subobject and external identifier of the log:

SET\_DESCRIPTOR

Name	Description
<b>Importing parameters</b>	
OBJECT	Object of the log
SUBOBJECT	Subobject of the log
EXTERNAL_ID	(Optional): External identifier of the log Default: ''
<b>Returning parameter</b>	
NEW_HEADER	Reference to current header object
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>OBJECT_NOT_ALLOWED: Access to the log object is not allowed
CX_BALI_INVALID_PARAMETER	The log object or subobject of the header doesn't exist

Set the expiry date and attributes:

SET\_EXPIRY

Name	Description
<b>Importing parameters</b>	
EXPIRY_DATE	Date when the log expires and can be deleted. If the parameter is initial, the expiry date remains unchanged; the default value of the expiry date is the creation date of the log + 7 days.
KEEP_UNTIL_EXPIRY	(Optional): If set: It is not allowed to delete the log before the expiry date Default: Not set
<b>Returning parameter</b>	
NEW_HEADER	Reference to current header object

Set the log context using a data dictionary structure:

SET\_CONTEXT

Name	Description
<b>Importing parameter</b>	
CONTEXT	Structure with the context data
<b>Returning parameters</b>	
NEW_HEADER	Reference to current header object
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_INVALID_PARAMETER	<ul style="list-style-type: none"><li>• Field CONTEXT is not based on a database table or structure which is defined in the data dictionary</li><li>• The total size of field CONTEXT is longer than 256 characters</li></ul>

Get all header values:

GET\_ALL\_VALUES

Name	Description
<b>Exporting parameters</b>	
OBJECT	Object of the log
SUBOBJECT	Subobject of the log
EXTERNAL_ID	External identifier of the log

Name	Description
EXPIRY_DATE	Date when the log expires and can be deleted
KEEP_UNTIL_EXPIRY	If set: It is not allowed to delete the log before the expiry date
CONTEXT_DATA	<p>Structure and content of the context. CONTEXT_DATA has the following fields:</p> <ul style="list-style-type: none"> <li>STRUCTURE_NAME: Name of the data dictionary structure which was used for the context data</li> <li>CONTENT: The content of the context data, stored in a character field with a length of 256 characters</li> </ul>

## 4.2.11.9.2.3.5 CL\_BALI\_MESSAGE\_SETTER (Interface IF\_BALI\_MESSAGE\_SETTER)

To add a new message to an application log an object of class *CL\_BALI\_MESSAGE\_SETTER* is used. The public interface of the instance methods is *IF\_BALI\_MESSAGE\_SETTER*. It contains the interface *IF\_BALI\_ITEM\_SETTER*.

### Public Attributes

Name	Description
CATEGORY	Category of the item
(from IF_BALI_ITEM_SETTER)	Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_MESSAGE

### Public Methods

Create a message class instance and set the message attributes:

CREATE (static)

Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc) Default: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
ID	Message ID
NUMBER	Message number
VARIABLE_1	(Optional): Message variable 1

Name	Description
VARIABLE_2	(Optional): Message variable 2
VARIABLE_3	(Optional): Message variable 3
VARIABLE_4	(Optional): Message variable 4
<b>Returning parameter</b>	
MESSAGE	Message object: A reference to interface IF_BALI_MESSAGE_SETTER

Create a message class instance. The message attributes are read from the fields of structure SY (like SY-MSGID):

CREATE\_FROM\_SY (static)

Name	Description
<b>Returning parameter</b>	
MESSAGE	Message object: A reference to interface IF_BALI_MESSAGE_SETTER

Create a message class instance. The message attributes are read from a structure of type BAPIRET2:

CREATE\_FROM\_BAPIRET2 (static)

Name	Description
<b>Importing parameter</b>	
MESSAGE_DATA	Message attributes (a structure of type BAPIRET2)
<b>Returning parameter</b>	
MESSAGE	Message object: A reference to interface IF_BALI_MESSAGE_SETTER

Set attributes of the message:

SET\_ATTRIBUTES

Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc) Default: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
ID	Message ID
NUMBER	Message number

Name	Description
VARIABLE_1	(Optional): Message variable 1
VARIABLE_2	(Optional): Message variable 2
VARIABLE_3	(Optional): Message variable 3
VARIABLE_4	(Optional): Message variable 4
<b>Returning parameter</b>	
NEW_MESSAGE	Reference to current message object

Set message attributes from the fields of structure SY (like SY-MSGID):

SET\_FROM\_SY

Name	Description
<b>Returning parameter</b>	
NEW_MESSAGE	Reference to current message object

Set message attributes from structure BAPIRET2:

SET\_FROM\_BAPIRET2

Name	Description
<b>Importing parameter</b>	
MESSAGE_DATA	Message attributes (a structure of type BAPIRET2)
<b>Returning parameter</b>	
NEW_MESSAGE	Reference to current message object

Set the level of detail of the item:

SET\_DETAIL\_LEVEL

Name	Description
<b>Importing parameter</b>	
DETAIL_LEVEL	Detail level of the item
	Allowed values: Number between '1' and '9' or ''
<b>Returning parameter</b>	
NEW_MESSAGE	Reference to current message object

Set the message context using a data dictionary structure:

## SET\_CONTEXT

Name	Description
<b>Importing parameters</b>	
CONTEXT	Structure with the context data
NEW_MESSAGE	Reference to the current message object
CX_BALI_INVALID_PARAMETER	<ul style="list-style-type: none"> <li>Field CONTEXT is not based on a database table or structure that is defined in the data dictionary</li> <li>The total size of field CONTEXT is longer than 256 characters</li> </ul>

Get all message values:

## GET\_ALL\_VALUES

Name	Description
<b>Exporting parameters</b>	
DETAIL_LEVEL	Detail Level of the item (number between '1' and '9' or '')
SEVERITY	Severity of the message ('Error', 'Warning', etc)
ID	Message ID
NUMBER	Message number
VARIABLE_1	(Optional): Message variable 1
VARIABLE_2	(Optional): Message variable 2
VARIABLE_3	(Optional): Message variable 3
VARIABLE_4	(Optional): Message variable 4
CONTEXT_DATA	Structure and content of the context. CONTEXT_DATA has the following fields: <ul style="list-style-type: none"> <li>STRUCTURE_NAME: Name of the data dictionary structure which was used for the context data</li> <li>CONTENT: The content of the context data. The content is stored in a character field with a length of 256 characters</li> </ul>

Check whether the message can pass an item filter:

## CHECK\_PASSING\_ITEM\_FILTER (from IF\_BALI\_ITEM\_SETTER)

Name	Description
<b>Importing parameter</b>	

Name	Description
ITEM_FILTER	Reference to the item filter that is being checked
<b>Returning parameter</b>	
FILTER_PASSED	If set, the message can pass the item filter

### ⓘ Note

If the severity of the message contains a value which is not allowed, it is changed to `IF_BALI_CONSTANTS=>C_SEVERITY_DEFAULT`. Allowed values of the severity can be found in interface `IF_BALI_CONSTANTS`.

If the detail level of the message contains a value which is not allowed, it is changed to `IF_BALI_CONSTANTS=>C_DETAIL_LEVEL_DEFAULT`. Allowed values of the detail level are a number between '1' and '9' and ''.

If the message attributes are set from structure `BAPIRET2`, only the fields `TYPE`, `ID`, `NUMBER`, `MESSAGE_V1`, `MESSAGE_V2`, `MESSAGE_V3` and `MESSAGE_V4` of the structure are considered.

## 4.2.11.9.2.3.6 CL\_BALI\_FREE\_TEXT\_SETTER (Interface IF\_BALI\_FREE\_TEXT\_SETTER)

To add a new free text to an application log, an object of class `CL_BALI_FREE_TEXT_SETTER` is used. The public interface of the instance methods is `IF_BALI_FREE_TEXT_SETTER`. It contains the interface `IF_BALI_ITEM_SETTER`.

### Public Attributes

Name	Description
CATEGORY	Category of the item
(from IF_BALI_ITEM_SETTER)	Contains fixed value: <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_FREE_TEXT</code>

### Public Methods

Create an instance of the free text class and set the text and the severity:

CREATE (static)

Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc)
	Default: <code>IF_BALI_CONSTANTS=&gt;C_SEVERITY_STATUS</code>

Name	Description
TEXT	Free text
<b>Returning parameter</b>	
FREE_TEXT	Free text object: A reference to interface IF_BALI_FREE_TEXT_SETTER
Set the free text and severity:	
SET_TEXT	
Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc)  Default: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
TEXT	Free text
<b>Returning parameter</b>	
NEW_FREE_TEXT	Reference to current free text object
Set the level of detail of the free text:	
SET_DETAIL_LEVEL	
Name	Description
<b>Importing parameter</b>	
DETAIL_LEVEL	Detail level of the free text  Allowed values: Number between '1' and '9' or ''
<b>Returning parameter</b>	
NEW_FREE_TEXT	Reference to current free text object
Set the free text context using a data dictionary structure:	
SET_CONTEXT	
Name	Description
<b>Importing parameter</b>	
CONTEXT	Structure with the context data
<b>Returning parameter</b>	
NEW_FREE_TEXT	Reference to the current free text object
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	

Name	Description
CX_BALI_INVALID_PARAMETER	<ul style="list-style-type: none"> <li>Field CONTEXT is not based on a database table or structure that is defined in the data dictionary</li> <li>The total size of field CONTEXT is longer than 256 characters</li> </ul>

Get all free text values:

GET\_ALL\_VALUES

Name	Description
<b>Exporting parameters</b>	
DETAIL_LEVEL	<p>Detail level of the free text</p> <p>Number between '1' and '9' or ''</p>
SEVERITY	Severity of the free text ('Error', 'Warning', etc)
TEXT	Content of the free text
CONTEXT_DATA	<p>Structure and content of the context. CONTEXT_DATA has the following fields:</p> <ul style="list-style-type: none"> <li>STRUCTURE_NAME: Name of the data dictionary structure which was used for the context data</li> <li>CONTENT: The content of the context data. The content is stored in a character field with a length of 256 characters</li> </ul>

Check whether the free text can pass an item filter:

CHECK\_PASSING\_ITEM\_FILTER (from IF\_BALI\_ITEM\_SETTER)

Name	Description
<b>Importing parameter</b>	
ITEM_FILTER	Reference to the item filter that is being checked
<b>Returning parameter</b>	
FILTER_PASSED	If set, the free text can pass the item filter

### ⓘ Note

If the severity of the free text contains a value which is not allowed, it is changed to IF\_BALI\_CONSTANTS=>C\_SEVERITY\_DEFAULT. Allowed values of the severity can be found in interface IF\_BALI\_CONSTANTS.

If the detail level of the free text contains a value which is not allowed, it is changed to IF\_BALI\_CONSTANTS=>C\_DETAIL\_LEVEL\_DEFAULT. Allowed values of the detail level are a number between '1' and '9' and ''.

## 4.2.11.9.2.3.7 CL\_BALI\_EXCEPTION\_SETTER (Interface IF\_BALI\_EXCEPTION\_SETTER)

To add a new exception to an application log, an object of class CL\_BALI\_EXCEPTION\_SETTER is used. The public interface of the instance methods is IF\_BALI\_EXCEPTION\_SETTER. It contains the interface IF\_BALI\_ITEM\_SETTER.

### Public Attributes

Name	Description
CATEGORY (from IF_BALI_ITEM_SETTER)	Category of the item Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_EXCEPTION

### Public Methods

Create an instance of the exception class. Set the reference of the ABAP exception which is stored in the exception item and its severity:

CREATE (static)

Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc) Default: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
EXCEPTION	Reference to ABAP exception class instance
<b>Returning parameter</b>	
EXCEPTION_OBJ	Exception object: A reference to interface IF_BALI_EXCEPTION_SETTER

Set the ABAP exception class instance and severity:

SET\_EXCEPTION

Name	Description
<b>Importing parameters</b>	
SEVERITY	(Optional): Severity of the message ('Error', 'Warning', etc) Default: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
EXCEPTION	Reference to ABAP exception class instance
<b>Returning parameter</b>	
NEW_EXCEPTION_OBJ	Reference to current application log exception object

Set the level of detail of the exception:

SET\_DETAIL\_LEVEL

Name	Description
<b>Importing parameter</b>	
DETAIL_LEVEL	Detail level of the exception Allowed values: Number between '1' and '9' or ''
NEW_EXCEPTION_OBJ	Reference to current application log exception object
<b>Returning parameter</b>	

Set the exception context using a data dictionary structure:

SET\_CONTEXT

Name	Description
<b>Importing parameter</b>	
CONTEXT	Structure with the context data
<b>Returning parameter</b>	
NEW_EXCEPTION_OBJ	Reference to the current application log exception object
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_INVALID_PARAMETER	<ul style="list-style-type: none"><li>Field CONTEXT is not based on a database table or structure that is defined in the data dictionary</li><li>The total size of field CONTEXT is longer than 256 characters</li></ul>

Get all exception values:

GET\_ALL\_VALUES

Name	Description
<b>Exporting parameters</b>	
DETAIL_LEVEL	Detail level of the exception (number between '1' and '9' or '')
SEVERITY	Severity of the exception ('Error', 'Warning', etc)
EXCEPTION	Reference to ABAP exception class instance

Name	Description
CONTEXT_DATA	<p>Structure and content of the context. CONTEXT_DATA has the following fields:</p> <ul style="list-style-type: none"> <li>STRUCTURE_NAME: Name of the data dictionary structure which was used for the context data</li> <li>CONTENT: The content of the context data. The content is stored in a character field with a length of 256 characters</li> </ul>

Check whether the exception can pass an item filter:

CHECK\_PASSING\_ITEM\_FILTER (from IF\_BALI\_ITEM\_SETTER)

Name	Description
<b>Importing parameter</b>	
ITEM_FILTER	Reference to the item filter that is being checked
<b>Returning parameter</b>	
FILTER_PASSED	If set, the exception can pass the item filter

#### ⓘ Note

If the severity of the exception contains a value which is not allowed, it is changed to IF\_BALI\_CONSTANTS=>C\_SEVERITY\_DEFAULT. Allowed values of the severity can be found in interface IF\_BALI\_CONSTANTS.

If the detail level of the exception contains a value which is not allowed, it is changed to IF\_BALI\_CONSTANTS=>C\_DETAIL\_LEVEL\_DEFAULT. Allowed values of the detail level are a number between '1' and '9' and ''.

## 4.2.11.9.2.3.8 CL\_BALI\_ITEM\_FILTER (Interface IF\_BALI\_ITEM\_FILTER)

The class CL\_BALI\_ITEM\_FILTER allows you to define an item filter. You can use it if it's necessary to restrict which items are added to an application log. The public interface of the instance methods is IF\_BALI\_ITEM\_FILTER.

### Public Methods

Create an instance of the filter class:

CREATE (static)

Name	Description
<b>Returning parameter</b>	
FILTER	Reference to the interface <code>IF_BALI_ITEM_FILTER</code>

Set a table of all severity values that pass the filter. It overwrites all previous filter settings related to the severity:

SET\_SEVERITY

Name	Description
<b>Importing parameter</b>	
SEVERITY_TABLE	Table with severity values (such as <code>Error</code> or <code>Warning</code> )
<b>Returning parameter</b>	
NEW_FILTER	Reference to the current filter object

#### ⓘ Note

- Valid values of the severity can be found in the interface `IF_BALI_CONSTANTS`
- If the parameter `SEVERITY_TABLE` doesn't contain an entry, all severity values pass the filter
- If the parameter `SEVERITY_TABLE` only contains invalid entries, no severity passes the filter

Apply the filter to a set of log item attributes and check whether it passes the filter:

APPLY\_FILTER

Name	Description
<b>Importing parameter</b>	
SEVERITY	Severity of the log item that is checked
<b>Returning parameter</b>	
FILTER_PASSED	If set, the set of attributes passes the item filter

## 4.2.11.9.2.3.9 IF\_BALI\_HEADER\_GETTER

If the header attributes of an application log are read, they are returned in an object which uses interface `IF_BALI_HEADER_GETTER`.

### Public Attributes

Name	Description
OBJECT	Object of the log

Name	Description
SUBOBJECT	Subobject of the log
EXTERNAL_ID	External identifier of the log
LOG_TIMESTAMP	UTC time stamp of the log (usually the creation time)
LOG_USER	Log user (usually the user who created the log)
EXPIRY_DATE	Date when the log expires and can be deleted
KEEP_UNTIL_EXPIRY	If set: It's not allowed to delete the log before the expiry date
NUMBER_ALL_ITEMS	Total number of items which are stored in the log
NUMBER_ABORT_ITEMS	Number of abort items which are stored in the log
NUMBER_ERROR_ITEMS	Number of error items which are stored in the log
NUMBER_WARNING_ITEMS	Number of warning items which are stored in the log
NUMBER_INFORMATION_ITEMS	Number of information items which are stored in the log
NUMBER_STATUS_ITEMS	Number of status items which are stored in the log
CONTEXT_STRUCTURE_NAME	Data dictionary name of the context structure

### Public Methods

Get description text of the log object in the logon language:

GET\_OBJECT\_DESCRIPTION

Name	Description
<b>Returning parameter</b>	
OBJECT_DESCRIPTION	Description of the object in the logon language

Get description text of log subobject in the logon language:

GET\_SUBOBJECT\_DESCRIPTION

Name	Description
<b>Returning parameter</b>	
SUBOBJECT_DESCRIPTION	Description of the subobject in the logon language

Get the context data of the header:

## GET\_CONTEXT

Name	Description
<b>Exporting parameter</b>	
CONTEXT	Structure that is filled with context data

## 4.2.11.9.2.3.10 IF\_BALI\_EXCEPTION\_GETTER

If an exception is read from an application log, an object instance of interface `IF_BALI_EXCEPTION_GETTER` is returned. It contains the interface `IF_BALI_ITEM_GETTER`.

### Public Attributes

Name	Description
CATEGORY (from <code>IF_BALI_ITEM_GETTER</code> )	Category of the item Contains fixed value: <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_EXCEPTION</code>
LOG_ITEM_NUMBER (from <code>IF_BALI_ITEM_GETTER</code> )	Serial number which is the position of the exception in the log
SEVERITY (from <code>IF_BALI_ITEM_GETTER</code> )	Severity of the exception ('Error', 'Warning', etc)
DETAIL_LEVEL (from <code>IF_BALI_ITEM_GETTER</code> )	Detail level of the exception (number between '1' and '9' or '')
TIMESTAMP (from <code>IF_BALI_ITEM_GETTER</code> )	UTC time stamp of the exception creation
CONTEXT_STRUCTURE_NAME (from <code>IF_BALI_ITEM_GETTER</code> )	Data dictionary name of the context structure
EXCEPTION_CLASS	Name of the ABAP exception class
EXCEPTION_ID_NAME	Name of the Text ID of the ABAP exception

### Public Methods

Get the message short text of the exception (the output of method `GET_TEXT` of the ABAP exception object):

`GET_MESSAGE_TEXT` (from interface `IF_BALI_ITEM_GETTER`)

Name	Description
<b>Returning parameter</b>	

Name	Description
MESSAGE_TEXT	Message short text of the exception in the logon language

Get the context data of the exception:

GET\_CONTEXT (from the interface IF\_BALI\_ITEM\_GETTER)

Name	Description
<b>Exporting parameter</b>	
CONTEXT	Structure that is filled with context data

## 4.2.11.9.2.3.11 IF\_BALI\_FREE\_TEXT\_GETTER

If a free text is read from an application log, an object instance of interface IF\_BALI\_FREE\_TEXT\_GETTER is returned. It contains the interface IF\_BALI\_ITEM\_GETTER.

### Public Attributes

Name	Description
CATEGORY (from IF_BALI_ITEM_GETTER)	Category of the item Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_FREE_TEXT
LOG_ITEM_NUMBER (from IF_BALI_ITEM_GETTER)	Serial number which is the position of the free text in the log
SEVERITY (from IF_BALI_ITEM_GETTER)	Severity of the free text ('Error', 'Warning', etc)
DETAIL_LEVEL (from IF_BALI_ITEM_GETTER)	Detail level of the free text (number between '1' and '9' or '')
TIMESTAMP (from IF_BALI_ITEM_GETTER)	UTC time stamp of the free text creation
CONTEXT_STRUCTURE_NAME (from IF_BALI_ITEM_GETTER)	Data dictionary name of the context structure

### Public Methods

Get the content of the free text:

GET\_MESSAGE\_TEXT (from interface IF\_BALI\_ITEM\_GETTER)

Name	Description
<b>Returning parameter</b>	
MESSAGE_TEXT	Content of the free text

Get the context data of the free text:

GET\_CONTEXT ( from the interface IF\_BALI\_ITEM\_GETTER)

Name	Description
<b>Exporting parameter</b>	
CONTEXT	Structure that is filled with context data

## 4.2.11.9.2.3.12 IF\_BALI\_ITEM\_GETTER

If an item like a message or exception is read from an application log, its object instance contains the interface IF\_BALI\_ITEM\_GETTER. So, interface IF\_BALI\_ITEM\_GETTER contains all attributes and methods which are available for all items which are read from the log.

### Public Attributes

Name	Description
CATEGORY	Category of the item Possible values: <ul style="list-style-type: none"><li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_MESSAGE: Item is a message</li><li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_FREE_TEXT: Item is a free text</li><li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_EXCEPTION: Item is an exception</li></ul>
LOG_ITEM_NUMBER	Serial number which is the position of the item in the log
SEVERITY	Severity of the item ('Error', 'Warning', etc)
DETAIL_LEVEL	Detail level of the item (number between '1' and '9' or ' ')
TIMESTAMP	UTC time stamp of the item creation
CONTEXT_STRUCTURE_NAME	Data dictionary name of the context structure

### Public Methods

Get the message text of the item. It is:

- The output of ABAP command **MESSAGE** for a message item
- The text of a free text item
- The output of method **GET\_TEXT** of the ABAP exception object for an exception item

**GET\_MESSAGE\_TEXT**

Name	Description
<b>Returning parameter</b>	
MESSAGE_TEXT	Text of the item message in the logon language

Get the context data of the item:

**GET\_CONTEXT**

Name	Description
<b>Exporting parameter</b>	
CONTEXT	Structure that is filled with context data

## 4.2.11.9.2.3.13 IF\_BALI\_ITEM\_SETTER

### Context

If a new item, such as a message or an exception, is added to an application log, its object instance contains the interface **IF\_BALI\_ITEM\_SETTER**.

### Public Attributes

Name	Description
CATEGORY	<p>Category of the item</p> <p>Possible values:</p> <ul style="list-style-type: none"> <li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_MESSAGE: Item is a message</li> <li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_FREE_TEXT: Item is a free text</li> <li>• IF_BALI_CONSTANTS=&gt;C_CATEGORY_EXCEPTION: Item is an exception</li> </ul>

## Public Methods

Check whether the item can pass an item filter:

CHECK\_PASSING\_ITEM\_FILTER

Name	Description
<b>Importing parameter</b>	
ITEM_FILTER	Reference to the item filter that is being checked
<b>Returning parameter</b>	
FILTER_PASSED	If set, the item can pass the item filter

## 4.2.11.9.2.3.14 IF\_BALI\_MESSAGE\_GETTER

If a message is read from an application log, an object instance of interface IF\_BALI\_MESSAGE\_GETTER is returned. It contains the interface IF\_BALI\_ITEM\_GETTER.

### Public Attributes

Name	Description
CATEGORY (from IF_BALI_ITEM_GETTER)	Category of the item Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_MESSAGE
LOG_ITEM_NUMBER (from IF_BALI_ITEM_GETTER)	Serial number which is the position of the message in the log
SEVERITY (from IF_BALI_ITEM_GETTER)	Severity of the message ('Error', 'Warning', etc)
DETAIL_LEVEL (from IF_BALI_ITEM_GETTER)	Detail level of the message (number between '1' and '9' or ' ')
TIMESTAMP (from IF_BALI_ITEM_GETTER)	UTC time stamp of the message creation
CONTEXT_STRUCTURE_NAME (from IF_BALI_ITEM_GETTER)	Data dictionary name of the context structure
ID	Message ID
NUMBER	Message number

Name	Description
VARIABLE_1	Message variable 1
VARIABLE_2	Message variable 2
VARIABLE_3	Message variable 3
VARIABLE_4	Message variable 4
COUNT	Count of cumulated messages

### Public Methods

Get the message text of the message (the output of ABAP command **MESSAGE**):

GET\_MESSAGE\_TEXT (from interface IF\_BALI\_ITEM\_GETTER):

Name	Description
<b>Returning parameter</b>	
MESSAGE_TEXT	Text of the message in the logon language

Get the context data of the message:

GET\_CONTEXT (from the interface IF\_BALI\_ITEM\_GETTER)

Name	Description
<b>Exporting parameter</b>	
CONTEXT	Structure that is filled with context data

## 4.2.11.9.3 How to Use the Fiori Reuse Library

You can use the [Reuse Library](#) to implement a reusable component that can display an application log.

For more information on how to use the Fiori reuse library in a freestyle app, see [Freestyle App Integration \[page 1603\]](#).

For more information on how to use the Fiori reuse library in a Fiori elements app, see [Fiori Elements Integration OData V2 \[page 1605\]](#).

## 4.2.11.9.3.1 Freestyle App Integration

You can implement the reusable component from the [Reuse Library](#) in a freestyle app in order to display an application log.

### Context

To be able to use the reusable components, you need to adapt the code that can be found in your view definition file and your controller file.

### Procedure

1. In the view definition file (`<MY_LOG_DISPLAY_VIEW.view.xml>`), the log display must be positioned on the page. Prepare a container control that will hold the log display component:

#### ↔ Sample Code

```
<!-- xmlns:core="sap.ui.core" -->
<core:ComponentContainer id="LogMessagesControlContainer" />
```

2. In the controller file (`<MY_LOG_DISPLAY_VIEW.controller.js>`), the relevant coding goes into the handler for the `onInit` event. Create the component and place it into the container control:

#### ⓘ Note

Make sure to replace the string `<MY_LOG_DISPLAY_VIEW>` with your own view's name!

#### ↔ Sample Code

```
var that = this;

this.getOwnerComponent().createComponent({
    usage: "ApplicationLogs",
    id: "LogMessagesControlComponent",
    settings: {
        "persistencyKey": "MY_LOG_DISPLAY_VIEW",
        "showHeader": false,
        "showFilterBar": false,
        "showAsTree": false
    }
}).then(function (oComp) {
    that.byId("LogMessagesControlContainer").setComponent(oComp);
    oComp.setLogHandle("<LogHandle>");
    oComp.refresh();
});
```

#### ⓘ Note

The component you embed makes use of the `SmartFilter` and the `SmartTable` control. Both controls enable a user to configure the components interactively, and also to store a current set of configuration

settings as a named variant. By providing a value for the `persistenceKey` parameter, you make sure that the variants that get created in your application become visible only to the users of your application, and not to all users of the component in all applications.

3. Adapt your `<manifest.json>` file and add the `sap.nw.core.applogs.lib.reuse.applogs` component:

#### ↔ Sample Code

```
"sap.ui5": {
  "dependencies": {
    "libs": {
      "sap.nw.core.applogs.lib.reuse": {
        "lazy": true
      }
    },
    "components": {}
  },
  "componentUsages": {
    "ApplicationLogs": {
      "name": "sap.nw.core.applogs.lib.reuse.applogs",
      "lazy": true
    }
  }
}
```

After your app was deployed successfully to an SAP BTP, ABAP environment system, the *BSP application* and the *SAP Fiori launchpad* app descriptor item will appear under your created package in Eclipse.



4. Now, you need to create a new *IAM App*. Follow the instructions described here: [Defining an IAM App for the Business Service \[page 826\]](#).
5. Once created, you need to maintain the *Application Log* OData Service to call your application log data. To do this, go to the *Service* tab and add the *Application Log OData Service* by naming the service type `odata v2` and add the following service name: `API_LOG_MANAGEMENT_SRV 0001`. Mind that all 13 spaces between `SRV` and `0001` are required. Now, go to the *Authorization* tab and maintain your *log objects / sub objects* for the authorization object `S_APPL_LOG`.
6. Finally, you need to create a new *Business Catalog*. Please follow the procedure described here: [Creating a Business Catalog \[page 827\]](#)

Having created the business catalog, you have successfully implemented a reusable component to display your application logs.

## 4.2.11.9.3.2 Fiori Elements Integration OData V2

You can implement the reusable component from the *Reuse Library* in a Fiori Elements app in order to display an application log.

### Context

In order to implement the reusable component in a Fiori Elements app, you first have to add the application log reuse component to your manifest.

### Procedure

1. Open the `manifest.json` file of the application and locate the `sap.ui5` section.
2. Add the `sap.nw.core.applogs.lib.reuse.smarttemplate` component:

#### Sample Code

```
"sap.ui5": {  
    "dependencies": {  
        "libs": {  
            "sap.nw.core.applogs.lib.reuse": {  
                "lazy": true  
            }  
        },  
        "components": {}  
    },  
    "componentUsages": {  
        "ApplicationLogs": {  
            "name": "sap.nw.core.applogs.lib.reuse.smarttemplate",  
            "lazy": true  
        }  
    }  
}
```

3. Next, you need to add the smarttemplate component of the application log reuse library to an `ObjectPage`. In the `manifest.json` file, you have to add the application log reuse component as an embedded component to your `ObjectPage`:

#### Sample Code

##### Snippet (embeddedComponents)

```
"embeddedComponents": {  
    "appLog": {  
        "id": "<enter an ID>",  
        "componentUsage": "ApplicationLogs",  
        "title": "<the name of the tab>",  
        "settings": {  
            "logHandle": "{LogHandle}",  
            "persistencyKey": "<your content>"  
        }  
    }  
}
```

```
}
```

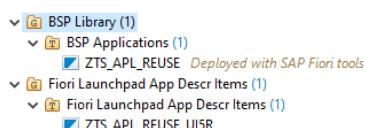
4. You can set the following parameters:

- **log handle**: The application log handle
- **Optional persistencyKey**: Set a personalization key which is then used to set app-specific variants for SmartTable. This parameter is optional. The default value is defined in the *Application Log* app. Make sure not to change the key value again, since users might lose their variants.

5. Here are some examples for binding:

- `logHandle: {myODataServiceLogHandle}`
- `logHandle: {path: 'myODataServiceLogHandle'}`

After your app was deployed successfully to an SAP BTP, ABAP environment system, the *BSP application* and the *SAP Fiori Launchpad* app descriptor item will appear under your created package in Eclipse.



6. Now, you need to create a new *IAM App*. Follow the instructions described here: [Defining an IAM App for the Business Service \[page 826\]](#).
7. Once created, you need to maintain the *Application Log* OData Service to call your application log data. To do this, go to the *Service* tab and add the *Application Log OData Service* by naming the service type *OData v2* and add the following service name: *API\_LOG\_MANAGEMENT\_SRV 0001*. Please make sure to add all 13 spaces in between *SRV* and *0001*. Now, go to the *Authorization* tab and maintain your *log objects / sub objects* for the authorization object *S\_APPL\_LOG*.
8. Finally, you need to create a new *Business Catalog*. Please follow the procedure described here: [Creating a Business Catalog \[page 827\]](#)

Having created the business catalog, you have successfully implemented a reusable component to display your application logs.

## 4.2.11.10 Data Archiving

Data archiving is used to extract data from the database, write it to archive files and delete the data from the database tables.

Data archiving is needed for data volume management. Usually, you want to remove data from the database which is not needed for your daily business anymore, but needs to be retained.

### Steps to implement a data archiving solution

1. Create a storage manager implementation to store the archive data to an external storage system [\[page 1607\]](#)
2. Select data and write it to archive files [\[page 1608\]](#)
3. Delete data from the database [\[page 1610\]](#)
4. Read archived data [\[page 1613\]](#)
5. Reload archived data into the database [\[page 1616\]](#)
6. Create a definition for archiving an object in ADT

## 7. Perform data archiving [page 1618]

For data archiving in ABAP Cloud the *Archive Development Kit* ( ADK ) provides released APIs or interfaces for all of these steps, which can be used in your implementation to archive content of your own tables.

### 4.2.11.10.1 Storage Manager Implementation

The storage manager integrates the data archiving runtime with the actual storage of the archiving files.

The storage manager will be processed in the different steps of the data archiving process. Within the storage manager implementation, you have the full flexibility to choose which storage will be integrated. Possible options to integrate external storages already exist in ABAP Cloud:

- [Via Cloud Connector for On-Premise integration](#)
- [Outbound Communication](#)

As a precondition for data archiving, you need to create a class which implements the interface `IF_ARCH_STORAGE_MANAGER`. This interface provides the following methods:

- The method `IF_ARCH_STORAGE_MANAGER~STORE_FILE` stores the archived data to a physical storage location outside of the SAP system. It will be called by the ADK write API and has the following parameters:

Parameter	Description
Importing: <code>IV_ARCHIVE_KEY</code>	Key of the archive file to be stored
Importing: <code>IV_DATA</code>	Content of the archive file (binary data)
Importing: <code>IV_LENGTH</code>	Length of the binary content
Returning: <code>RV_DOCUMENT_ID</code>	Document identifier which can be used to access the archived data. The document ID needs to be returned by the storage manager and will be stored in a link table in the SAP system.

- The method `IF_ARCH_STORAGE_MANAGER~GET_BYTESTREAM` retrieves parts of the binary content of an archive file via range request. It will be called by an ADK API to read the archived data and has the following parameters:

Parameter	Description
Importing: <code>IV_DOCUMENT_ID</code>	Document identifier
Importing: <code>IV_FROM_OFFSET</code>	Data object address
Importing: <code>IV_LENGTH</code>	Length of the binary content
Exporting: <code>EV_DATA</code>	Content of the archive file (binary data)
Exporting: <code>EV_ACT_LENGTH</code>	Length of the byte stream which is returned

- The method `IF_ARCH_STORAGE~RETRIEVE_FILE` retrieves the entire binary content of an archive file.

Parameter	Description
Importing: <i>IV_DOCUMENT_ID</i>	Document identifier
Returning: <i>RV_DATA</i>	Content of the archive file (binary data) which will be returned by the storage manager

- The method *IF\_ARCH\_STORAGE\_MANAGER~DELETE FILE* deletes the archive file from the external storage system.

Parameter	Description
Importing: <i>IV_DOCUMENT_ID</i>	Document identifier

#### ⓘ Note

If any errors appear during any of the operations of the storage manager, please use the exception class *CX\_ARCH\_STORAGE\_MANAGER*.

### 4.2.11.10.2 Select Data and Write It to Archive Files

The runtime to write archived data to an external storage should be implemented in a central ABAP class. This ABAP class must register the interface *IF\_ARCH\_WRITE\_API* to be able to be selected in an archiving object definition in ADT. This ABAP class can be [performed within an application job \[page 1618\]](#).

The purpose of the ABAP class is the selection of application data which will be archived from the database. By using the ADK write API, this application data can be handed over to create a data object. A data object can contain entries of different tables that have been listed in the definition of the archiving object in ADT. Typically, these are all tables belonging to one business object instance, but there is no technical restriction. The application is free to group the tables according to their needs.

As soon as the data object is completed, the object will be transferred as an archiving file to the related storage manager implementation to store data in the related external storage.

The following table shows the write API to collect the data that will be archived:

CL_ARCH_WRITE_API		
Method	Parameter	Description
<i>GET_INSTANCE</i>	Importing: <i>IV_ARCHIVING_OBJECT</i>	Name of the archiving object which will be used
Static method which returns an instance of the write API via parameter <i>RO_WRITE_INSTANCE</i>	Importing: <i>IV_TESTMODE</i>	<i>True</i> : no storage takes place <i>False</i> : Archive content is written, and storage is requested
	Returning: <i>RO_WRITE_INSTANCE</i>	Instance of the write API for further processing

The following table shows the interface for writing archived data:

IF_ARCH_WRITE_API		
Method	Parameter	Description
<a href="#">OPEN_DATA_OBJECT</a>		
	Opens a new data object for the data of the business object instance	
<a href="#">PUT_DATA_RECORDS</a>	Importing: <i>IV_TABLE_NAME</i>	Name of the table which will be archived
Writes the data for a specific table to the open data object	Importing: <i>IT_RECORDS</i>	Records of the table
<a href="#">CLOSE_DATA_OBJECT</a>	Exporting: <i>EV_OBJECT_OFFSET</i>	Offset at which the data of the data object starts (can be used to build application owned indices for the archived data)
Closes the data object and writes it to the archive file	Exporting: <i>EV_ARCHIVE_KEY</i>	Archive key of the archive file to which the data was written (can be used to build application owned indices for the archived data)
	Exporting: <i>EV_ARCHIVING_SESSION</i>	Sessions of the archiving object to be used
<a href="#">FINALIZE</a>		
Finalizes processing and closes the archiving session		

Typical flow of the archiving write process:

1. Create an instance of the write API and open the archiving write session by using method `CL_ARCH_WRITE_API=>GET_INSTANCE`.
2. Select the data from your application DB tables.
3. For each data object you want to archive, call method `OPEN_DATA_OBJECT`.
4. For each table you want to archive entries from, call method `PUT_DATA_RECORDS`.
5. As soon as all tables of your data object are passed to the API, call method `CLOSE_DATA_OBJECT`.
6. Either start with the next data object (step 3) or finish the archiving write session by calling method `FINALIZE`.

### ↔ Sample Code

```
CONSTANTS: lc_object TYPE search_object VALUE 'MY_OBJECT'.
" Replace with the name of your ADT archiving object
" select flight data (leading table) from the database (replace table names
with your names of your own tables)
  SELECT * FROM my_header_table
    WHERE carrid      IN @lt_range_carrid
      AND connid      IN @lt_range_connid
      AND fldate      IN @lt_range_fldate
    INTO TABLE @DATA(lt_flights).
" open a new archiving session to archive data
TRY.
  DATA(lo_write) =
cl_arch_write_api=>get_instance( iv_archiving_object =
lc_object
                                iv_testmode = lv_test ).
```

```

LOOP AT lt_flights ASSIGNING FIELD-SYMBOL(<ls_flight>).
" select data of dependent tables (replace table names with your names of
your own tables)
    INSERT <ls_flight> INTO lt_flights.
    SELECT * FROM my_item_table1
        WHERE carrid      = @<ls_flight>-carrid
          AND connid      = @<ls_flight>-connid
          AND fldate       = @<ls_flight>-fldate
    INTO TABLE @DATA(lt_bookings).
    SELECT * FROM my_item_table2
        WHERE carrid      = @<ls_flight>-carrid
          AND connid      = @<ls_flight>-connid
          AND fldate       = @<ls_flight>-fldate
    INTO TABLE @DATA(lt_tickets).
    SELECT * FROM my_item_table3
        WHERE carrid      = @<ls_flight>-carrid
          AND connid      = @<ls_flight>-connid
          AND fldate       = @<ls_flight>-fldate
    INTO TABLE @DATA(lt_invoices).
" open new ADK data object
lo_write->open_data_object( ).
" write data of header table
lo_write->put_data_records( iv_table_name = '<MY_HEADER_TALBE>'
it_records = lt_sfflight_arch ).
CLEAR lt_sfflight_arch.
" write data of depending tables
lo_write->put_data_records( iv_table_name = '<MY_ITEM_TABLE1>'
it_records = lt_bookings ). " bookings
CLEAR lt_sfflight_arch.
lo_write->put_data_records( iv_table_name = '<MY_ITEM_TABLE2>'
it_records = lt_tickets ). " tickets
CLEAR lt_sfflight_arch.
lo_write->put_data_records( iv_table_name = '<MY_ITEM_TABLE3>'
it_records = lt_invoices ). " invoices
CLEAR lt_sfflight_arch.
" write data object into the archive file
lo_write->close_data_object( ).
ENDLOOP.
" finalize and end writing
lo_write->finalize( ).
```

### 4.2.11.10.3 Delete Data from the Database

The runtime to delete archived data from database should be implemented in a central ABAP class. This ABAP class must register interface `IF_ARCH_DELETE_API` to be able to be selected in an archiving object definition in ADT. This ABAP class can be [performed within an application job \[page 1618\]](#).

There are different ways to delete archived data from the database:

- Using a central delete API to delete all table entries of a data object
- Using a central delete API to delete specified tables of a data object
- Using a central delete API to get all table entries of a data object to perform an own deletion from the database

#### `CL_ARCH_DELETE_API` (API for deleting archived data from the database)

Method	Parameter	Description
--------	-----------	-------------

**CL\_ARCH\_DELETE\_API (API for deleting archived data from the database)**

<b>GET_INSTANCE</b>	Importing: <i>IV_ARCHIVING_OBJECT</i>	Name of the archiving object which will be used
Returns an instance of the delete API. Reference to interface <i>IF_ARCH_DELETE_API</i>	Importing: <i>IV_ARCHIVE_KEY</i>	Archive key for which data should be deleted
	Importing: <i>IV_TESTMODE</i>	<i>True</i> : no storage takes place <i>False</i> : Archive content is written, and storage is requested
	Returning: <i>RO_DELETE_INSTANCE</i>	Instance of the delete API
<b>GET_FILES_TO_DELETE</b>	Importing: <i>IV_ARCHIVING_OBJECT</i>	Name of the archiving object which will be used
Returns a table with archive keys which have not yet been processed by deletion	Importing: <i>IT_DATE_SELECTION</i>	Range of dates on which archive files have been created
	ReturnImporting: <i>IT_USER_SELECTION</i>	Range of users which have created archive files
	Returning: <i>RT_ARCHIVE_FILE_ATTRIBUTES</i>	Table of archive keys which fulfill the selection criteria for dates and users

**IF\_ARCH\_DELETE\_API (Interface for deleting archived data from the database)**

<i>Method</i>	<i>Parameter</i>	<i>Description</i>
<b>GET_NEXT_OBJECT</b>	Exporting: <i>EV_OBJECT_OFSET</i>	Offset at which the data of the data object starts. This can be used to build application-owned indices for the archived data.
Reads the next data object for the archived data	Exporting: <i>EV_ARCHIVE_KEY</i>	Archive key of the archive file to which the data was written. This can be used to build application-owned indices for the archived data.
	Exporting: <i>EV_ARCHIVING_SESSION</i>	Sessions of the archiving object which will be used
	Exporting: <i>EV_END_OF_FILE</i>	<i>True</i> : All data contained in the files is processed, and the application should finalize the deletion <i>False</i> : The application should continue processing the next data object
<b>GET_DATA_RECORDS</b>	Importing: <i>IV_RECORD_STRUCTURE</i>	Name of the table for which archived data should be returned
	Exporting: <i>ET_DATA_RECORDS</i>	Records of the table
<b>DELETE_DATA_FOR_TABLE</b>	Importing: <i>IV_TABLE_NAME</i>	Name of the table for which archived data should be deleted
Deletes the data for a specific table from the database	Importing: <i>IT_RECORDS</i>	Records of the table to be deleted

## IF\_ARCH\_DELETE\_API (Interface for deleting archived data from the database)

---

### DELETE\_DATA\_FOR\_OBJECT

Deletes all table entries contained in the data object

---

### FINALIZE

Finalizes processing

---

Typical flow of the archiving delete process:

1. Determine files which are still available for deletion by using the method `CL_ARCH_DELETE_API=>GET_FILES_TO_DELETE`, with filter options on creating user and creation date of the archive file.
2. Create an instance of the delete API and open the archiving delete session for one of the selected archive files by using method `CL_ARCH_DELETE_API=>GET_INSTANCE`.
3. Open the next data object with the method `GET_NEXT_DATA_OBJECT`. If the method returns `EV_END_OF_FILE = TRUE ('X')`, continue with step 6. Optional (can be used for reading archived data via index access): Update the index table with the data object archived.
4. Delete the data of the object from your database tables either by:
  - Use of the method `DELETE_DATA_FOR_OBJECT`
  - Use of the method `DELETE_DATA_FOR_TABLE`. Call this method once for each table.
  - Use of the method `GET_DATA_RECORDS`. Use the result to perform your own `DELETE` statements.
5. Continue with the next data object. See step 3.
6. Finish the archiving delete session by calling the method `FINALIZE`.

### ↔ Sample Code

```
CONSTANTS: lc_object TYPE if_arch_api_types=>ty_object_name VALUE 'MY_OBJECT'.
DATA: lv_test TYPE abap_bool VALUE abap_true,
lt_data TYPE STANDARD TABLE OF my_index_table,
lv_arkey TYPE search_archive_key.
DATA: ls_index TYPE my_index_table.

" get newest undeleted file
  DATA(lt_files) =
cl_arch_delete_api=>get_files_to_delete( iv_archiving_object =
lc_object ).           SORT lt_files BY creation_date DESCENDING creation_time
DESCENDING.               READ TABLE lt_files ASSIGNING FIELD-SYMBOL(<ls_file>) INDEX
1.
  DATA(lo_delete) = cl_arch_delete_api=>get_instance
( iv_archiving_object =
lc_object
    iv_archive_key = lv_arkey
                                iv_testmode = lv_test ).
  " read next data object from archive file into internal memory
  DO.
    lo_delete->get_next_data_object( IMPORTING ev_end_of_file =
DATA(lv_end_of_file)
                                    ev_archive_key =
DATA(lv_archive_key)
                                    ev_object_offset =
DATA(lv_offset) .
    IF lv_end_of_file = abap_true.
      EXIT.
    ENDIF.
```

```

        " optional: Create customer owned index for single document
access during read
        " read archived data for table MY_HEADER_TABLE
        lo_delete->get_data_records( EXPORTING iv_record_structure =
'MY_HEADER_TABLE'
                                IMPORTING et_data_records =
lt_data ).
        " map flight data to your index table MY_INDEX_TABLE
IF lv_test <> abap_true.
LOOP AT lt_data ASSIGNING FIELD-SYMBOL(<ls_data>).
    MOVE-CORRESPONDING <ls_data> TO ls_index.
    ls_index-archive_key = lv_archive_key.
    ls_index-archive_offset = lv_offset.
    INSERT my_index_table FROM @ls_Index.
ENDLOOP.
ENDIF.
" delete the data from the data base
lo_delete->delete_data_for_object( ).
ENDDO.
COMMIT WORK.
lo_delete->finalize( ).
```

Example of a customer-owned index table for archived flights:

#### ↔ Sample Code

```

@EndUserText.label : 'Archiving Index: Flights'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #ALLOWED
define table my_index_table {
@EndUserText.label : 'Client'
key mandt : abap.clnt not null;
@EndUserText.label : 'Airline Code'
key carrid : abap.char(3) not null;
@EndUserText.label : 'Flight Connection Number'
key connid : abap.numc(4) not null;
@EndUserText.label : 'Flight date'
key fldate : abap.dats not null;
archive_key : sarch_archive_key;
archive_offset : sarch_offset;
```

## 4.2.11.10.4 Read Archived Data

Applications can use the ADK read API to read the data which is contained in an archive file stored in an external storage system.

There are two possible ways to access the data:

- Read the data for entire archive files sequentially
- Read the data by index access

To optimize the reading access on archive files, an index solution can be established in the application. For an index-based read access, the archive key and the offset of a data object are needed. ADK API provides this information within methods `IF_ARCH_DELETE_API->GET_NEXT_DATA_OBJECT` and `IF_ARCH_READ_API->GET_NEXT_DATA_OBJECT`.

A typical index-based solution needs a transparent table containing columns for application attributes, a column for the archive key and a column for the offset. The archive key is the ID of the archive file, which contains the application attribute. The offset is the offset in the archive file at which the data for the application attribute starts. Usually, the initial construction of an archive index will be established by [deleting archived data from the database \[page 1610\]](#). In case the index should be constructed at a later point in time, the ADK read API can be used to determine archive key and offset (returned by method `IF_ARCH_READ_API->GET_NEXT_DATA_OBJECT`).

#### `CL_ARCH_READ_API` (API to read archived data)

Method	Parameter	Description
<code>GET_FILES_TO_READ</code>	Importing: <code>IT_DATE_SELECTION</code>	Range on dates on which archive files have been created
Selection of archiving sessions and files	Importing: <code>IT_USER_SELECTION</code>	Range on users which have created archive files
	Importing: <code>IT_ARCHIVING_OBJECT</code>	Archiving object which will be used
	Return-ing: <code>RT_ARCHIVING_FILE_ATTRIBUTES</code>	Archiving sessions' attributes
<code>GET_INSTANCE</code>	Importing: <code>IV_ARCHIVING_OBJECT</code>	Name of the archiving object which will be used
Returns an instance of the delete API. Reference to interface <code>IF_ARCH_READ_API</code>	Importing: <code>IV_ARCHIVING_SESSION</code>	Sessions of the archiving object which will be used
	Importing: <code>IT_ARCHIVE_KEY</code>	Archive key for which data should be read
	Returning: <code>RO_READ_INSTANCE</code>	Instance of the read API for further processing

#### `IF_ARCH_READ_API` (Interface to read archived data)

Method	Parameter	Description
<code>GET_DATA_RECORDS</code>	Importing: <code>IV_RECORDS_STRUCTURE</code>	Name of the structure of all data records in the table
Read records by structure from the current data object	Exporting: <code>EV_DATA_RECORDS</code>	Table containing the data records
<code>GET_NEXT_DATA_OBJECT</code>	Exporting: <code>EV_ARCHIVE_KEY</code>	Archive key according to archive management
Read data object from the archive file	Exporting: <code>EV_END_OF_FILE</code>	Boolean values:  <code>TRUE</code> (= 'X')  <code>FALSE</code> (= '')
	Exporting: <code>EV_ARCHIVING_SESSION</code>	Sessions of the archiving object which will be used
	Exporting: <code>EV_OBJECT_OFFSET</code>	Offset of data object in the archive

#### `CLOSE`

Open archive files will be closed

As mentioned before, there are two different ways to access the data. A typical procedure for the sequential read access would be as follows:

1. Determine files which are available for evaluation by using the method `CL_ARCH_READ_API=>GET_FILES_TO_READ`, with filter options on creating user and creation date of the archive file.
2. Create an instance of the read API and open the archiving read session for one of the selected archive files by using the method `CL_ARCH_READ_API=>GET_INSTANCE`.
3. Open the next data object with the method `GET_NEXT_DATA_OBJECT`. If the method returns `EV_END_OF_FILE = TRUE ('X')`, continue with step 6.
4. Read the data with the method `GET_DATA_RECORDS`.
5. Continue with the next data object. See step 3.
6. Finish the read operation by calling method `CLOSE`.

A typical flow of the index-based read access would be as follows:

1. Determine the archive key and the offset of the data object by selecting the relevant entry from your index table.
2. Create an instance of the read API and open the archiving read session for the data object by using the method `CL_ARCH_READ_API=>GET_INSTANCE_BY_OFFSET`.
3. Read the data by using the method `GET_DATA_RECORDS`.
4. Finish the read operation by calling method `CLOSE`.

An example code for the index-based read operation:

#### ↳ Sample Code

```
CONSTANTS: lc_object TYPE if_arch_api_types=>ty_object_name VALUE 'MY_OBJECT'.
DATA: lt_data TYPE STANDARD TABLE OF my_item_table.
DATA: lt_index TYPE STANDARD TABLE OF my_index_table.
      " let's assume, we want to read the archived bookings for flight AA/0040
      on 20010110
      TRY.
         " get archive key and offset for the given flight from the index table
         SELECT * FROM my_index_table WHERE carrid = 'AA' AND connid = '0040'
AND fldate = '20010110'
         INTO TABLE @lt_index.
         SORT lt_index BY archive_key DESCENDING.
         READ TABLE lt_index ASSIGNING FIELD-SYMBOL(<ls_index>) INDEX 1.
         IF NOT <ls_index> IS ASSIGNED.
            RETURN.
         ENDIF.
         " open a read instance for the archive key and offset found
         DATA(lo_read) = cl_arch_read_api=>get_instance_by_offset
( iv_archiving_object = lc_object
                           iv_archive_key =
<ls_index>-archive_key
                           iv_object_offset = <ls_index>-
archive_offset ).
         " get all archived booking entries for this flight
         lo_read->get_data_records( EXPORTING iv_record_structure =
'MY_ITEM_TABLE1'
                           IMPORTING et_data_records = lt_data ).
         " close the read instance
         lo_read->close( ).
```

## 4.2.11.10.5 Reload Archived Data into the Database

Reloading is optional and should only be used in exceptional cases. The application must decide if a reload is necessary or not.

In the standard process, a reloading of archived data is not required, as archived data and direct access to individual data objects can be performed by using the [Read API \[page 1613\]](#).

There are two cases which need to be distinguished:

1. Data must be reloaded shortly after deleting data from the database within the archiving process. An example for this case would be if selection criteria were set wrongly for the archiving process.
2. Reload of archived data after a long period of time. In this case it must be ensured that references to and within the reloaded data are still available in database. Furthermore, it needs to be guaranteed that no table key conflicts exist. Since this is not a real error, it's strongly recommended not to reload the data in this case.

The runtime to reload archived data from an external storage system back into the database should be implemented in a central ABAP class. This ABAP class has to register interface `IF_ARCH_RELOAD_API` to be able to be selected in an archiving object definition in ADT.

### `CL_ARCH_RELOAD_API` (API to reload archived data)

Method	Parameter	Description
<code>GET_INSTANCE</code>	Importing: <code>IV_ARCHIVING_OBJECT</code>	Name of the archiving object which will be used
Returns an instance of the delete API. Reference to interface <code>IF_ARCH_RELOAD_API</code>	Importing: <code>IV_ARCHIVING_SESSION</code>	Sessions of the archiving object which will be used
	Importing: <code>IV_TESTMODE</code>	Reload program runs in test mode
	Returning: <code>RO_RELOAD_INSTANCE</code>	Instance of the reload API for further processing
<code>GET_SESSIONS_TO_RELOAD</code>	Importing: <code>IT_DATA_SELECTION</code>	Range on dates on which archive files have been created
Read a data object from an archive file	Importing: <code>IT_USER_SELECTION</code>	Range on users which have created archive files
	Importing: <code>IV_ARCHIVING_OBJECT</code>	Archiving object which will be used
	Returning: <code>RT_SESSION_ATTRIBUTES</code>	Archiving sessions' attributes

### `IF_ARCH_RELOAD_API` (Interface to reload archived data)

Method	Parameter	Description
<code>GET_DATA_RECORDS</code>	Importing: <code>IV_RECORDS_STRUCTURE</code>	Name of a structure of all data records in the table
Read records by structure from the current data object	Exporting: <code>ET_DATA_RECORDS</code>	Table containing the data records
<code>GET_NEXT_DATA_OBJECT</code>	Exporting: <code>EV_ARCHIVE_KEY</code>	Archive key according to archive management
Read data object from the archive file		

## IF\_ARCH\_RELOAD\_API (Interface to reload archived data)

Exporting: <i>EV_END_OF_FILE</i>	Boolean values: <i>TRUE</i> (= 'X') <i>FALSE</i> (= '')
Exporting: <i>EV_ARCHIVING_SESSION</i>	Sessions of the archiving object which will be used
Exporting: <i>EV_OBJECT_OFFSET</i>	Offset of data object in the archive
<b><i>RELOAD_DATA_FOR_OBJECT</i></b>	
<i>RELOAD_DATA_FOR_TABLE</i> Reloading the data of an internal table	Importing: <i>IV_TABLE_NAME</i> DDIC structure name of an internal table Tables: <i>IT_TABLE_NAME</i>
<b><i>FINALIZE</i></b>	
Open archive files will be closed	

A typical flow of the archiving reload process would be as follows:

1. Determine archiving sessions which are still available for reload by using the method `CL_ARCH_RELOAD_API=> GET_SESSIONS_TO_RELOAD`, with filter options on creating user and creation date of the archive file.
2. Create an instance of the reload API and open the archiving reload session for one of the selected archiving sessions by using the method `CL_ARCH_RELOAD_API=>GET_INSTANCE`.
3. Open the next data object via method `GET_NEXT_DATA_OBJECT`. If the method returns `EV_END_OF_FILE = TRUE ('X')`, continue with step 6 to call the method `FINALIZE`.
4. Reload the data of the object from the external storage system back into your database tables, either by:
  - Use of the method `RELOAD_DATA_FOR_OBJECT`
  - Or use of the method `RELOAD_DATA_FOR_TABLE`. This method needs to be called once for each table.
  - Use of the method `GET_DATA_RECORDS`. With this method, the result needs to be used to perform your own `INSERT` statements.
5. Continue with the next data object. See step 3.
6. Finish the archiving reload session by calling the method `FINALIZE`.

An example coding snippet:

### ↔ Sample Code

```
CONSTANTS: lc_object TYPE if_arch_api_types=>ty_object_name VALUE 'MY_OBJECT'.
DATA:      lv_test TYPE abap_bool VALUE abap_true,
lt_sessions TYPE STANDARD TABLE OF my_header_table,
lv_session TYPE search_session_number.

      " get newest complete archiving session
      DATA(lt_sessions) =
cl_arch_reload_api=>get_sessions_to_reload(           iv_archiving_object =
lc_object ).                                     SORT lt_sessions BY creation_date DESCENDING creation_time
DESCENDING.                                         READ TABLE lt_sessions ASSIGNING FIELD-SYMBOL(<ls_session>)
INDEX 1.
```

```

        DATA(lo_reload) = cl_arch_reload_api->get_instance
( iv_archiving_object =    lc_object
                           iv_archiving_session =
lv_session
                           iv_testmode =
lv_test ).          " read next data object from archive file into internal memory
DO.
    lo_reload->get_next_data_object( IMPORTING ev_end_of_file =
DATA(lv_end_of_file)
                           ev_archive_key =
DATA(lv_archive_key)
                           ev_object_offset =
DATA(lv_offset) ).      IF lv_end_of_file = abap_true.
    EXIT.
ENDIF.
" optional: remove
" read archived data for table ZADK_SFLIGHT
lo_reload->get_data_records( EXPORTING iv_record_structure =
'MY_HEADER_TABLE'
                           IMPORTING et_data_records =
lt_data ).      " reload the data into the data base
lo_reload->reload_data_for_object( ).      " optional: delete data from customer owned index
DELETE FROM <my_index_table>
    WHERE archive_key = @lv_archive_key.
ENDDO.
COMMIT WORK.
lo_reload->finalize( ).
```

## 4.2.11.10.6 Archiving Classes

Archiving classes are used to archive object instances of reuse objects together with the business object instances in your application to which the reuse object instance belongs. If your application includes such reuse objects, and if SAP offers a released archiving class for the reuse object, you can easily integrate the archiving class into your archiving object:

- Enhance the archiving object by adding the archiving class to your archiving object in the ADT editor
- Enhance the write class by registering the object instance of the reuse object via a released method belonging to the archiving class
- Use the released read functionality belonging to the archiving class to read the data of the reuse object

## 4.2.11.10.7 Perform Data Archiving

Productive data archiving is a mass data processing and should therefore always be performed in the background, decoupled from the transactional business process. For background processing, regular performing and monitoring of data archiving runs, the *Application Job* framework is recommended to use. For that, an application job catalog entry with at least one application job template is required for each data archiving step:

- Write data to external storage
- Delete data from custom database tables

The application job catalog entry contains the implementation of the data archiving step: it uses the released corresponding ADK runtime API. For logging, the application log in the context of an application job can be used. Related monitoring is embedded in the application job apps.

All APIs for data archiving provide a test mode parameter. If this parameter is set, no changes are performed on the database and no data is stored via the storage manager. Applications themselves need to expose a test mode parameter to the user. They need to take care that no database changes are performed in their part of the write, delete, and reload ABAP classes in case the test mode option is chosen.

## Related Information

[Application Jobs \[page 1501\]](#)

[Application Logs \[page 1536\]](#)

### 4.2.11.11 Output Management

[Printing \[page 1619\]](#)

Find out how to create a print queue item API.

[Print Forms \[page 1620\]](#)

ADS Rendering in ABAP environment

[Emailing \[page 1627\]](#)

#### 4.2.11.1.1 Printing

Find out how to create a print queue item API.

You can use the method `CREATE_QUEUE_ITEM_BY_DATA` with the class `CL_PRINT_QUEUE_UTILS` to create a print queue item of a queue. One print queue item has one main document and can have either no or a variable amount of attachments. The method `CREATE_QUEUE_ITEM_BY_DATA` has the following importing parameters:

- `IV_QNAME`: Name of the print queue
- `IV_PRINT_DATA`: Print document contained in xstring format
- `IV_NAME_OF_MAIN_DOC`: Name of the main document
- `IV_NUMBER_OF_COPIES`: Number of copies
- `IV_PAGES`: Number of pages
- `IT_ATTACHMENT_DATA`: List of attachments within the structure. This includes the name and the attachment data in xstring

If a print queue item has been created successfully, it returns the ID of the item with the parameter `RV_ITEMID`. In case of any errors, you can find the error messages in the exporting parameter `EV_ERR_MSG`.

## • Example

```
CLASS zcl_pq_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun .
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_pq_test IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: l_print_data TYPE xstring.
    l_print_data = |...|.
    DATA(lv_qitem_id) = cl_print_queue_utils=>create_queue_item_by_data
      EXPORTING
        iv_qname          = '...'
        iv_print_data     = l_print_data
        iv_name_of_main_doc = '...'.
    out->write( lv_qitem_id ).
  ENDMETHOD.
ENDCLASS.
```

### 4.2.11.11.2 Print Forms

ADS Rendering in ABAP environment

The *Adobe Document Services* (ADS) can render Adobe XML Forms (XFA) to PDF documents and other print formats. The application sends the data and the form template using a secure connection and the service returns the rendered document. The rendered documents can be used by your applications for the screen output or for the output into the print queue. For more information, see [Maintain Print Queues \[page 2715\]](#).

For more information, see [Connecting the SAP Forms Service to ABAP](#).

#### 4.2.11.11.2.1 Runtime API for ADS Rendering Calls

The CL\_FP\_ADS\_UTIL class provides the ABAP Runtime API for *Adobe Document Services* (ADS) rendering calls. It contains the following *Public* methods to be used for the corresponding functions:

Public Methods

Method	Description
RENDER_PDF	Rendering PDF
RENDER_4_PQ	Rendering for Print Queue

The PDF rendering process on ADS creates the PDF file to be printed (or print output format) using the following both formats:

- XDP form template file
- Data for the form filling in XML format, for example, data from the [RAP Data Services for Print Forms \[page 1623\]](#).

The XDP form template file is a description of the form layout design in Adobe XFA format. The supported data binding type is an XML data schema.

The [Adobe LiveCycle Designer for SAP solutions](#) is the tool for visual design XDP form templates from Adobe Inc. It can be downloaded using the [Install Additional Software](#) app, which contains the download link to the SAP ONE Support Launchpad.

The data XML file contains the data that corresponds to the used data schema in the XDP form template.

## Rendering PDF

The RENDER\_PDF method calls the ADS to render print PDF using the XDP form file and the XML data file. It has the following importing/exporting parameters:

Importing Parameters

Parameter	Description
IV_XML_DATA	XML data
IV_XDP_LAYOUT	Adobe XDP form template
IV_LOCALE	Locale for rendering the language: language_COUNTRY, for example en_US
IS_OPTIONS	PDF rendering parameters (optional)

Exporting Parameters

Parameter	Description
EV_PDF	PDF rendering result
EV_PAGES	Number of pages
EV_TRACE_STRING	Trace string

### ↔ Sample Code

```
TRY.  
*Render PDF  
    cl_fp_ads_util->render_pdf( EXPORTING iv_xml_data      = lv_xml_data  
                                iv_xdp_layout     = lv_xdp  
                                iv_locale        = 'de_DE'  
                                is_options       = ls_options  
                                IMPORTING ev_pdf      = ev_pdf  
                                ev_pages         = ev_pages
```

```

        ev_trace_string = ev_trace_string
    ).

CATCH cx_fp_ads_util INTO lx_fp_ads_util.

ENDTRY.

```

## Rendering for Print Queue

The RENDER\_4\_PQ method calls the ADS to render in the print output format required for the target print queue. The support of the *Print Definition Language* (PDL) print queue format will be taken from print queue properties.

### Importing Parameters

Parameter	Description
IV_XML_DATA	XML data
IV_XDP_LAYOUT	Adobe XDP form template
IV_LOCALE	Locale for rendering the language: language_COUNTRY, for example en_US
IV_PQ_NAME	Print Queue name
IS_OPTIONS	PDL rendering parameters (optional)

### Exporting Parameters

Parameter	Description
EV_PDL	PDF rendering result
EV_PAGES	Number of pages
EV_TRACE_STRING	Trace string

### Sample Code

```

TRY.
*Render in Print Queue Format
  cl_fp_ads_util=>render_4_pq( EXPORTING iv_xml_data      = lv_xml_data
                                 iv_xdp_layout     = lv_xdp
                                 iv_locale        = 'en_US'
                                 iv_pq_name       = 'PQ1'
                                 is_options        = ls_options
                               IMPORTING ev_pdf          = ev_pdf
                                         ev_pages        = ev_pages
                                         ev_trace_string = ev_trace_string
                                         ).
  CATCH cx_fp_ads_util INTO lx_fp_ads_util.
ENDTRY.

```

## 4.2.11.2.2 RAP Data Services for Print Forms

The XML data for print form rendering can be obtained from the RAP Business Services. The data model can be defined just as well as the [RAP Service Definition](#). For more information, see [Service Definition](#).

The CL\_FP\_FDP\_SERVICES class provides the ABAP API to receive the following:

- data in xml format for the use in print forms
- XML Schema Definition (XSD) used for the design of print forms in the [Adobe LiveCycle Designer](#).

### Initiate Business Data Reader

To supply the rendering operation with business data, a reuse service can be applied. Beforehand, a service definition needs to be created.

#### ⓘ Note

The service definition doesn't need a binding for RAP Data Services for print forms.

1. Initiate the reuse utility by specifying the service definition: `data(lo_fdp_api) = cl_fp_fdp_services->get_instance(`FP_FDP_AUNIT_SO`).`
2. Specify the key parameters. This is required to read the data for a specific item. The reuse utility offers the following functionality to retrieve all key fields for the service definition: `data(lt_keys) = lo_fdp_api->get_keys( ).`
3. Once you have retrieved the array with key values, assign the keys that fit to the document you want to output: `lt_keys[ name = 'UUID' ]-value = 'FA163EE47BDD1ED9A682A2E6F1ECF696'.`

#### ↴ Sample Code

```
DATA(lo_fdp_api) = cl_fp_fdp_services->get_instance(`FP_FDP_SERVICE_EX1`).
DATA(lt_keys)      = lo_fdp_api->get_keys( ).
lt_keys[ name = 'UUID' ]-value = 'FA163EE47BDD1ED9A682A2E6F1ECF696'.
```

### Retrieve XML Data from Service Definition

The READ\_TO\_XML method retrieves the data structure of the service definition for a specific item. It has the following Importing parameters:

Importing Parameters

Parameter	Definition
IT_SELECT	Key parameters

Parameter	Definition
IV_LANGUAGE (Optional)	Overwrites the locale language (useful if you want to control how dynamic values (such as unites) are resolved)

The READ\_TO\_XML method has the following returning parameters:

Returning Parameters

Parameter	Definition
RV_XML	Resulting XML as XSTRING

#### ↳ Sample Code

```
TRY.
  DATA(lt_keys)      = lo_fdp_api->get_keys(  ).
  lt_keys[ name = 'UUID' ]-VALUE = 'FA163EE47BDD1ED9A682A2E6F1ECF696'.
  DATA(lv_data) = lo_fdp_api->read_to_xml( lt_keys ).
  CATCH cx_fp_fdp_error INTO DATA(lo_exception).
ENDTRY.
```

## Retrieve XML Schema Definition from Service Definition

The GET\_XSD method retrieves the XML Schema Definition (XSD) for the service definition. It has the following returning parameters:

Returning Parameters

Parameter	Definition
RV_XML	Resulting XSD as XSTRING

#### ↳ Sample Code

```
TRY.
  DATA(lv_xml)      = lo_fdp_api->get_xsd(  ).
  CATCH cx_fp_fdp_error INTO DATA(lo_exception).
ENDTRY
```

## Read XML Data for a Service Definition Using Draft Functionality

By default, for draft-enabled services, the `IsActiveEntity` property is exported implicitly, which describes if the active or the draft data should be read. Because this property is always false, by default, it needs to be overwritten during a service call to read the active data from the service definition.

You can use the following sample code. Mind that you need to initiate the business data reader first (see above).

#### ↳ Sample Code

```
TRY.  
    DATA(lt_keys)      = lo_fdp_api->get_keys( ).  
    lt_keys[ name = 'UUID' ]-VALUE = 'FA163EE47BDD1ED9A682A2E6F1ECF696'.  
    DATA(lv_data) = lo_fdp_api->read_to_xml(  
        it_select      = lt_keys  
        it_select_add = VALUE if_fp_fdp_api->tt_select_keys( name =  
            'IsActiveEntity' value = 'X' data_type =  
            'ABAP_BOOL' )  
    ).  
    CATCH cx_fp_fdp_error INTO DATA(lo_exception).  
ENDTRY.
```

### 4.2.11.11.2.3 PDF Merger API

You can use a class-based API to merge the content of different PDF files into a single PDF file. The result file contains all pages of the source files in the sequence of the original files.

If you want to merge PDF files, you require an instance of class CL\_RSPO\_PDF\_MERGER. To create this instance, class CL\_RSPO\_PDF\_MERGER provides the method CREATE\_INSTANCE. After using this method, the following two steps are necessary:

- With method ADD\_DOCUMENT you can add the data of a PDF document to the list of files you want to merge. You should call this method for each source file.
- With method MERGE\_DOCUMENTS you can start the merge process. The method returns the data of the merged document.

#### ↳ Sample Code

```
...  
    " Create an instance of the PDF merger class  
    DATA(l_merger) = cl_rspon_pdf_merger->create_instance( ).  
    " Add the data of the first PDF document to the list of files which shall  
be merged  
    l_merger->add_document( l_data_of_first_pdf ).  
    " Add the data of the second PDF document  
    l_merger->add_document( l_data_of_second_pdf ).  
TRY.  
    " Merge both documents and receive the result  
    DATA(l_merged_PDF) = l_merger->merge_documents( ).  
    CATCH cx_rspon_pdf_merger INTO DATA(l_exception).  
        " Add a useful error handling here  
    ENDTRY.  
...  
...
```

If you try to merge damaged documents, or documents that don't fulfill the requirements of the PDF merger, an exception of type CX\_RSPO\_PDF\_MERGER is raised. The exception object contains the following attributes that contain more information regarding the problem:

- DOCUMENT\_INDEX: The index of the PDF document that was responsible for the exception

- `ERROR_CODE`: The internal error code of the exception
- `ERROR_TEXT`: The description text of the error code

The PDF merger has the following restrictions:

- It can only handle PDF files.
- It can't handle encrypted PDF files. Usually, you can check whether a PDF file is encrypted by checking the [Security](#) section of the file properties. Another possibility is that you open the file with a text editor and search for the key word **Encrypt**, which marks an encrypted file.
- It can't handle damaged PDF files. For example, if parts of the file are missing, or if some bytes were changed, the file can't be processed. PDF files are usually binary data, so any conversion like a code page conversion damages the file.
- Further restrictions can be found in SAP note [2264208](#).

## Related Information

[Class and Interface of the PDF Merger API \[page 1626\]](#)

## 4.2.11.11.2.3.1 Class and Interface of the PDF Merger API

Class `CL_RSPO_PDF_MERGER` uses interface `IF_RSPO_PDF_MERGER`. Find out which public methods it contains.

## Context

The following public methods exist in interface `IF_RSPO_PDF_MERGER`:

### CREATE\_INSTANCE (static)

Create an instance of the PDF merger class.

Name	Description
<b>Returning parameter</b>	
<code>MERGER_INSTANCE</code>	PDF merger instance: A reference to interface <code>IF_RSPO_PDF_MERGER</code>

### ADD\_DOCUMENT

Add a PDF document to the list of files that you want to merge.

Name	Description
<b>Importing parameter</b>	
DOCUMENT	Data of the PDF document that you want to merge (type XSTRING)

## MERGE\_DOCUMENTS

Merge all PDF documents into a single PDF file.

Name	Description
<b>Returning parameter</b>	
MERGED_DOCUMENT	Data of the merged document (type XSTRING)
<b>Exceptions</b>	
CX_RSPO_PDF_MERGER	Exceptions of the PDF merger

## Related Information

[PDF Merger API \[page 1625\]](#)

### 4.2.11.11.3 Emailing

#### 4.2.11.11.3.1 Sending Mails Using SMTP

Send mails using the Simple Message Transfer Protocol (SMTP).

You can send mails with the Simple Message Transfer Protocol (SMTP) using a mail server connected via the *SAP Business Technology Platform (SAP BTP)*, *Cloud Connector* or use a publicly available SMTP server.

For more information, see [Configure Access Control \(TCP\)](#)

## Application Programming Interface (API)

Use the `CL_BCS_MAIL_MESSAGE` class to create and send mails. You can specify sender and recipient. Furthermore, you can add textual body parts using class `CL_BCS_MAIL_TEXTPART` or binary attachments (e.g. PDF documents) represented by class `CL_BCS_MAIL_BINARYPART`, respectively.

With the send method of the API, the mail is sent via the mail server configured in the destination using the cloud connector. Sending is performed synchronously.

## ↔ Sample Code

```
try.
    data(lo_mail) = cl_bcs_mail_message->create_instance( ).
    lo_mail->set_sender( 'noreply@yourcompany.com' ).
    lo_mail->add_recipient( 'recipient1@yourcompany.com' ).
    lo_mail->add_recipient( iv_address = 'recipient2@yourcompany.com' iv_copy
= cl_bcs_mail_message->cc ).
    lo_mail->set_subject( 'Test Mail' ).
    lo_mail->set_main( cl_bcs_mail_textpart->create_text_html( '<h1>Hello</
h1><p>This is a test mail.</p>' ) ).
    lo_mail->add_attachment( cl_bcs_mail_textpart->create_text_plain(
        iv_content      = 'This is a text attachment'
        iv_filename     = 'Text_Attachment.txt'
    )).
    lo_mail->add_attachment( cl_bcs_mail_textpart->create_instance(
        iv_content      = '<note><to>John</to><from>Jane</from><body>My nice
XML!</body></note>'
        iv_content_type = 'text/xml'
        iv_filename     = 'Text_Attachment.xml'
    )).
    data(lo_mail_status_monitor) = lo_mail->send( ).

    lo_mail_status_monitor->get_email_status(
        importing
        es_mail_status      = data(ls_mail_status)
        et_recipients_statuses = data(lt_recipients_statuses) ).

    catch cx_bcs_mail into data(lx_mail).
        "handle exceptions here
endtry.
```

## Bodyparts in a Message

Each mail message consists of one or more body parts (MIME parts). A body part consists of the text or binary content, the type and additional attributes like the filename for an attachment. A main body part (`SET_MAIN`) must be set and additional body parts can be set as attachments (`ADD_ATTACHMENT`) or alternatives (`ADD_MAIN_ALTERNATIVE`). Attachments are usually files that are attached to the mail. Alternative body parts are different representations of the main part of the email. For example, a mail can have an HTML representation and clients that can only display text or plain text will see this alternative body part.

The factory methods `CREATE_TEXT_PLAIN`, `CREATE_TEXT_HTML` of class `CL_BCS_MAIL_TEXTPART` can be used for the most common body parts. If a body part is to be sent with a different content-type, the `CREATE_INSTANCE` method can be used and a content-type can be passed.

## Sending Emails Asynchronously

In RESTful application programming (RAP) or in other transactional contexts, you may need to send an email asynchronously from time to time. This can be due to the performance, since external communication can take a while, or because you want to send an email when the `COMMIT WORK` statement is triggered at the end of a successful transaction.

An email can be sent asynchronously with the method `SEND_ASYNC`. See the following example:

#### ↔ Sample Code

```
try.  
    data(lo_mail) = cl_bcs_mail_message->create_instance( ).  
    lo_mail->set_sender( 'noreply@yourcompany.com' ).  
    lo_mail->add_recipient( 'recipient1@yourcompany.com' ).  
    "...  
    lo_mail->send_async( ).  
    catch cx_bcs_mail into data(lx_mail).  
        "handle exceptions here  
endtry.
```

After calling the `SEND_ASYNC` method, a background process is triggered. The process can be monitored in the Fiori app [Monitor Email Transmissions \[page 2727\]](#).

#### ⚠ Caution

The process must call `COMMIT WORK` at the end of a transaction, otherwise the background process will not start and the email status will remain Waiting in the Monitor Email Transmissions app.

## Email Status Monitoring

The status of an email can be monitored using the Fiori app [Monitor Email Transmission](#). Additionally, it is possible to implement a monitoring instance directly in the backend for each email request. This monitoring instance, which is an implementation of the interface `IF_BCS_MAIL_STATUS_MONITOR`, provides information about the email status as well as the status of each recipient, including the SMTP response. The `send()` method provides the same statuses as exporting parameters.

To obtain this monitoring instance, you can either receive it as the return parameter of the methods `send()` or `send_async()`, or create a new instance using the factory method `create_mail_status_monitor()` of the class `CL_BCS_MAIL_MESSAGE`.

### 4.2.11.3.2 Configuration of the System Email Outbound

Configure the email system using class `CL_BCS_MAIL_SYSTEM_CONFIG`. This configuration provides additional functionality to an SMTP email configuration.

#### Configuring the Email Outbound Check

You can restrict the email outbound by checking the sender and recipient addresses. The check is executed when the email is sent. If the address doesn't match with the sender or recipient domains or addresses on the allowlist, the email will not be sent and is not displayed in the app.

In addition, you can set a default sender address.

## Activating the Email Outbound Check

Use class `CL_BCS_MAIL_SYSTEM_CONFIG` and method `SET_ADDRESS_CHECK_ACTIVE()` to activate the email outbound check. The importing parameter `ABAP_TRUE` activates the check, `ABAP_FALSE` deactivates it. The functionality is inactive per default.

## Configuring the Email Outbound Allow Tables

To restrict the email outbound, two tables can be filled with allowed domains or email addresses. The first table contains allowed recipients (filled with method `ADD_ALLOWED_RECIPIENT_DOMAINS()`). The second table contains allowed senders (filled with method `ADD_ALLOWED_SENDER_DOMAINS()`). The check can be used for both the sender address and recipient address(es). If either all sender or recipient addresses should be allowed, add a `\*` to the corresponding allow domain table.

The configuration must be done as system customizing. It's not allowed to set the configuration in a session together with any email outbound.

## Configuring the Default Sender

In addition to the allowed domains, you can define a default sender. This default sender will be used as sender at anytime when no sender has been defined for an outgoing email. Please keep in mind that the default sender must be a valid email address. If no default sender is maintained and no sender is set in the outgoing email, the sender will be set to `do.not.reply@(system-default-domain)`.

### ⓘ Note

You can define only one default sender per system.

## Application Programming Interface (API)

To maintain the allowed domains and the default sender, further methods are provided to read or delete the current table data:

### ↔ Sample Code

```
DATA(config_instance) = cl_bcs_mail_system_config->create_instance( ).  
DATA recipient_domains TYPE cl_bcs_mail_system_config->tyt_recipient_domains.  
DATA sender_domains TYPE cl_bcs_mail_system_config->tyt_sender_domains.  
recipient_domains = VALUE #( ( 'recipient1domain.com' )  
    ( 'recipient2domain.com' ) ).  
sender_domains = VALUE #( ( 'sender1domain.com' ) ( 'sender2domain.com' ) ).  
"Add allowed domains  
TRY.  
    config_instance->set_address_check_active( abap_true ).  
    config_instance->add_allowed_recipient_domains( recipient_domains ).  
    config_instance->add_allowed_sender_domains( sender_domains ).  
    config_instance->modify_default_sender_address( iv_default_address =  
        'DefaultSender@yourcompany.com'  
            iv_default_name = 'Default  
Sender' ).  
    CATCH cx_bcs_mail_config INTO DATA(write_error).  
        "handle exception  
ENDTRY.  
"Read allowed domains  
DATA(allowed_recipient_domains) = config_instance->read_allowed_recipient_domains( ).  
DATA(allowed_sender_domains) = config_instance->read_allowed_sender_domains( ).  
config_instance->read_default_sender_address(  
    IMPORTING  
        ev_default_sender_address = DATA(default_sender_address)  
        ev_default_sender_name = DATA(default_sender_name) ).
```

```

"Delete allowed domains
TRY.
    config_instance->delete_allowed_rec_domains( allowed_recipient_domains ).
    config_instance->delete_allowed_sender_domains( allowed_sender_domains ).
    config_instance-
>delete_default_sender_addr( 'DefaultSender@yourcompany.com' ).
    CATCH cx_bcs_mail_config INTO DATA(deletion_error).
        "handle exception
ENDTRY.

```

## Configuring the Email Expiry Date

The expiry date of email send attempts can be defined using method `SET_DAYS_UNTIL_MAIL_EXPIRES()`. The default value for mail expiry is 30 days. After this time, sent and failed emails attempts will be deleted and no longer visible within the *Monitor Email Transmission* app.

If the `DELETE_DAYS_UNTIL_MAIL_EXPIRES()` method is used, the expiry date is set back to 30 days.

### 4.2.11.11.3.3 Email Address Check

The class `CL_MAIL_ADDRESS` implements the interface `IF_MAIL_ADDRESS` and allows the user to check if it contains a syntactic valid email address in the input string according to the standard RFC 5322.

#### Create an Email Address instance

Method	Description
<code>Create_instance</code>	Provides an email address instance.
<code>validate</code>	Validates whether a string is a syntactic valid mail address (according to RFC 5322 Standard).

#### Methods

##### ↔ Sample Code

```

Try.
    Data(lo_mail_address) =
    cl_mail_address->create_instance( iv_address_string = 'testmail@ok.com' ).
    Catch cx_bcs_mail into data(lx_err).
        "exception handling
    endtry.

Data(lv_address_valid) = lo_mail_address->validate( ).
```

## 4.2.11.12 Units of Measurement

Many business applications use units of measurement in their business processes. To standardize these processes, you need a central maintenance of units and related dimensions. Beside that, there's a business need for conversion between different units.

We provide a subset of common standardized units, dimensions, and ISO codes for use as predelivered content. In addition, you need to define customer-owned units and dimensions in customer applications.

### How are Units and Dimensions Linked?

Units are related to a dimension. In each dimension, a unit is defined as an SI unit (International System of Units). This is the basis for the conversion from one unit to another.

### Warnings

Changing a unit of measurement (UoM) may have an impact on your applications, especially on the critical fields.

UoMs are used across applications. An initial set of UoMs is delivered in the system. You can adapt UoMs as per your requirements. Be aware that by adding, changing, or deleting the customizing, you are changing the initial set of UoMs delivered in the system.

#### ⚠ Caution

Pay special attention if UoMs are already saved in transactional data of productive business processes. Changes of the fields marked as critical can result in serious inconsistencies.

We strongly recommend that you do **not** delete any dimensions, ISO codes, and units that are already defined in the system.

### Critical Fields

Critical Fields

For...

Critical Fields

Dimensions

*SI Unit*

The *SI Unit* is the base unit of a dimension for conversion. By changing the base unit, the conversion will be performed on a different base unit defined.

For...	Critical Fields
Units of Measurement	<ul style="list-style-type: none"><li>• Numerator</li><li>• Denominator</li><li>• Exponent</li><li>• Additive Constant</li><li>• ISO Code</li><li>• Primary Code</li></ul> <div style="border: 1px solid #e0e0e0; padding: 10px; margin-top: 10px;"><p><b>⚠ Caution</b></p><p>Numerator, denominator, exponent, and additive constant define the conversion factor with respect to the SI units. Any changes of the conversion factor will change the output result of a conversion between units under the same dimension.</p></div> <div style="border: 1px solid #e0e0e0; padding: 10px; margin-top: 10px;"><p><b>⚠ Caution</b></p><p>Changing the ISO code and primary code may have an impact on applications. For example, ISO codes are used as normed measurement unit IDs for electronic data interchange (EDI). If no ISO code is assigned to a unit, you may encounter issues in EDI.</p></div>

## Related Information

[Maintaining Dimensions \[page 1633\]](#)

[Maintaining Units of Measurement \[page 1640\]](#)

[Conversion Functions for Units of Measurement \[page 1649\]](#)

### 4.2.11.12.1 Maintaining Dimensions

Class `CL_UOM_DIM_MAINTENANCE` provides methods for maintaining a dimension.

#### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

For more information, see the following:

- [Creating a Dimension \[page 1634\]](#)
- [Changing a Dimension \[page 1636\]](#)

- [Deleting a Dimension \[page 1638\]](#)
- [Reading a Dimension \[page 1639\]](#)

## 4.2.11.12.1.1 Creating a Dimension

Use method `CREATE` to create a new dimension.

 **Caution**

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

### Import Parameters

Parameter Name	Field Name	Value Help
DIM_CRE_TS		Structure for creating a dimension
	DIMID	Dimension key
	TXDIM	Description of the dimension key
	LENGTH	Length exponent of the dimension
	MASS	Mass exponent of the dimension
	TIME	Time exponent of the dimension
	CURRENT	Electric current exponent of the dimension
	TEMPERATURE	Temperature exponent of the dimension
	MOLE_QTY	Mole quantity exponent of the dimension
	LUMINOSITY	Light exponent of the dimension

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error <b>'X':</b> Save error

### ⓘ Note

Class exception CX\_UOM\_ERROR is raised to check the integrity of the data import parameters.

### ↴ Sample Code

```
CLASS zcl_uom_dimension_create_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_dimension_create_test IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.

    DATA(lo_dim) = cl_uom_dim_maintenance=>get_instance( ).
    DATA(ls_dim) = VALUE cl_uom_dim_maintenance=>ty_dim_cre_ts( dimid =
'ZNEWDI'                                         txdim = 'New
Dimension'                                         mass   = 89 ).

    TRY.
      lo_dim->create( EXPORTING dim_cre_ts = ls_dim
                        IMPORTING error      = DATA(lv_error) ).

      CATCH cx_uom_error INTO DATA(lo_error).
        out->write( |Exception raised| ).
        out->write( lo_error->get_text( ) ).
    ENDTRY.
    IF lv_error = abap_true.
      out->write( |Error occurred while saving the data in the database| ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.
```

## 4.2.11.12.1.2 Changing a Dimension

Use method `UPDATE` to change an existing dimension.

### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

## Import Parameters

Parameter Name	Field Name	Value Help
DIMID		Dimension key
DIM_UPD_TS		Structure for updating a dimension
	TXDIM	Description of the dimension key
	SI_UNIT	Base unit of a dimension
	LENGTH	Length exponent of the dimension
	MASS	Mass exponent of the dimension
	TIME	Time exponent of the dimension
	CURRENT	Electric current exponent of the dimension
	TEMPERATURE	Temperature exponent of the dimension
	MOLE_QTY	Mole quantity exponent of the dimension
	LUMINOSITY	Light exponent of the dimension

### ⚠ Caution

Be aware that all the fields belonging to the structure will be updated.

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error <b>'X':</b> Save error

### ⓘ Note

Class exception CX\_UOM\_ERROR is raised to check the integrity of the data import parameters.

### ↴ Sample Code

```
CLASS zcl_uom_dimension_update_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.

  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_dimension_update_test IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA(lo_dim) = cl_uom_dim_maintenance=>get_instance( ).
    TRY.
      lo_dim->read( EXPORTING dimid      = 'ZNEWDI'
                     IMPORTING dim_st = DATA(ls_dim) ).
      DATA(ls_upd_dim) = VALUE cl_uom_dim_maintenance=>ty_dim_upd_ts(
        BASE CORRESPONDING #( ls_dim )
          txdim = 'Update Dimension'
          mass  = 88
          length = 89 ).
      lo_dim->update( EXPORTING dimid      = 'ZNEWDI'
                     dim_upd_ts = ls_upd_dim
                     IMPORTING error      = DATA(lv_error) ).
    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( |Exception raised| ).
      out->write( lo_error->get_text( ) ).
    ENDTRY.
    IF lv_error = abap_true.
      out->write( |Error occurred while updating the data in the database| ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.
```

### 4.2.11.12.1.3 Deleting a Dimension

Use method `DELETE` to delete a dimension.

#### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

### Import Parameters

Parameter Name	Value Help
DIMID	Dimension key

### Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error <b>'X':</b> Save error

#### ⓘ Note

Class exception `CX_UOM_ERROR` is raised to check the integrity of the data import parameters.

#### ↔ Sample Code

```
CLASS zcl_uom_dimension_delete_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_dimension_delete_test IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.

    DATA(lo_dim) = cl_uom_dim_maintenance=>get_instance( ).

    TRY.
      lo_dim->delete( EXPORTING dimid = 'ZNEWDI'
                      IMPORTING error = DATA(lv_error) ).
    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( |Exception raised| ).
```

```

        out->write( lo_error->get_text( ) ).
ENDTRY.
IF lv_error = abap_true.
  out->write( |Error occurred while deleting the data from the
database| ).
ENDIF.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.12.1.4 Reading a Dimension

Use method `READ` to read a dimension.

### Import Parameters

Parameter Name	Value Help
DIMID	Dimension key

### Export Parameters

Parameter Name	Field Name	Value Help
DIM_ST		Structure for reading a dimension
	DIMID	Dimension key
	TXDIM	Description of the dimension key
	SI_UNIT	Base unit of a dimension
	SI_TXT	Description of the base unit
	LENGTH	Length exponent of the dimension
	MASS	Mass exponent of the dimension
	TIME	Time exponent of the dimension
	CURRENT	Electric current exponent of the dimension
	TEMPERATURE	Temperature exponent of the dimension

Parameter Name	Field Name	Value Help
	MOLE_QTY	Mole quantity exponent of the dimension
	LUMINOSITY	Light exponent of the dimension

### ⓘ Note

Class exception CX\_UOM\_ERROR is raised if no dimension is found.

### ↴ Sample Code

```
CLASS zcl_uom_dimension_read_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_dimension_read_test IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.

    DATA(lo_dim) = cl_uom_dim_maintenance=>get_instance( ).

    TRY.
      lo_dim->read( EXPORTING dimid = 'AAAADL'
                     IMPORTING dim_st = DATA(ls_dim) ).
    CATCH cx_uom_error.
    ENDTRY.

    out->write( ls_dim-DIMID ).
    out->write( ls_dim-txdim ).

  ENDMETHOD.

ENDCLASS.
```

## 4.2.11.12.2 Maintaining Units of Measurement

Class CL\_UOM\_MAINTENANCE provides methods for maintaining units of measurement.

### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

For more information, see the following:

- [Creating a Unit of Measurement \[page 1641\]](#)
- [Changing a Unit of Measurement \[page 1643\]](#)
- [Deleting a Unit of Measurement \[page 1645\]](#)
- [Reading a Unit of Measurement \[page 1647\]](#)

## 4.2.11.12.2.1 Creating a Unit of Measurement

Use method CREATE to create a unit of measurement.

### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

### Import Parameters

Parameter Name	Field Name	Value Help
UNIT_DIMID		Dimension key
UNIT_INT		Internal unit of measurement
UNIT_CRE_TS		Structure for creating a unit of measurement
	COMMERCIAL	Commercial/external measurement unit format
	TECHNICAL	Technical measurement unit format
	DEC_ROUND	Number of decimal places to which this measurement unit should be rounded for conversion
	NUMERATOR	Numerator for conversion to SI unit
	DENOMINATOR	Denominator for conversion into SI unit
	EXPONENT	Base ten exponent for conversion to SI unit
	CONSTANT	Additive constant for conversion to SI unit

Parameter Name	Field Name	Value Help
	DEC_DISP	Number of decimal places with which this measurement unit is displayed
	ISOCODE	ISO code for measurement units. An ISO code can be assigned to several internal measurement units of a dimension.
	PRIMARY	Unit of measure flagged as a primary unit for an ISO code  Space: Not primary  'X': Set as primary
	TEXT	Description of a unit of measurement
	LONG_TEXT	Long description of a unit of measurement

## Export Parameters

Parameter Name	Value Help
ERROR	Space: No error  'X': Save error

### ⓘ Note

Class exception cx\_uom\_error is raised to check the integrity of the data import parameters.

### ↗ Sample Code

```

CLASS zcl_uom_unit_create_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_unit_create_test IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.

    DATA(lo_uom) = cl_uom_maintenance=>get_instance( ).
    DATA(ls_unit) = VALUE cl_uom_maintenance=>ty_uom_cre_ts( commercial =
      'ZYX'
  
```

```

'ZYX'
'1'
'1'
'3'
'Create Unit' ).

TRY.
    lo_uom->create( EXPORTING unit_dimid = 'AAAADL'
                      unit_int   = 'ZYX'
                      unit_cre_ts = ls_unit
                      IMPORTING error      = DATA(lv_error) ).

    CATCH cx_uom_error INTO DATA(lo_error).
        out->write( | Exception raised | ).
        out->write( lo_error->get_text( ) ).
ENDTRY.
IF lv_error = abap_true.
    out->write( |Error occurred while saving the data in the database| ).
ENDIF.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.12.2.2 Changing a Unit of Measurement

Use method `UPDATE` to change a unit of measurement.

### ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

## Import Parameters

Parameter Name	Field Name	Value Help
UNIT		Internal unit of measurement
UNIT_UPD_TS		Structure for updating a unit of measurement
<b>⚠ Caution</b> Be aware that all the fields belonging to the structure will be updated.		

Parameter Name	Field Name	Value Help
	COMMERCIAL	Commercial/external measurement unit format
	TECHNICAL	Technical measurement unit format
	DEC_ROUND	Number of decimal places to which this measurement unit should be rounded for conversion
	NUMERATOR	Numerator for conversion to SI unit
	DENOMINATOR	Denominator for conversion into SI unit
	EXPONENT	Base ten exponent for conversion to SI unit
	CONSTANT	Additive constant for conversion to SI unit
	DEC_DISP	Number of decimal places with which this measurement unit is displayed
	ISOCODE	ISO code for measurement units. An ISO code can be assigned to several internal measurement units of a dimension.
	PRIMARY	Unit of measure flagged as a primary unit for an ISO code  <b>Space:</b> Not primary  <b>'X'</b> : Set as primary
	TEXT	Description of a unit of measurement
	LONG_TEXT	Long description of a unit of measurement

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error  <b>'X'</b> : Save error

## ⓘ Note

Class exception `CX_UOM_ERROR` is raised to check the integrity of the data import parameters.

## ↴ Sample Code

```
CLASS zcl_uom_unit_update_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_unit_update_test IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA(lo_uom) = cl_uom_maintenance=>get_instance( ).
  TRY.
    lo_uom->read( EXPORTING unit      = 'ZYX'
                  IMPORTING unit_st = DATA(ls_unit) ).  

    DATA(ls_upd_unit) = VALUE cl_uom_maintenance=>ty_uom_upd_ts(
      BASE CORRESPONDING #( ls_unit )
        commercial      = 'ZYA'
        technical       = 'ZYA'
        denominator    = '1'
        numerator      = '1'
        dec_disp       = '5'
        long_text      = 'Updates Unit'
        text           = 'Upd Unit' ).  

    lo_uom->update( EXPORTING unit      = 'ZYX'
                  unit_upd_ts = ls_upd_unit
                  IMPORTING error     = DATA(lv_error) ).  

    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( |Exception raised| ).  

      out->write( lo_error->get_text( ) ).  

  ENDTRY.  

  IF lv_error = abap_true.
    out->write( |Error occurred while updating the data in the database| ).  

  ENDIF.
ENDMETHOD.
ENDCLASS.
```

### 4.2.11.12.2.3 Deleting a Unit of Measurement

Use method `DELETE` to delete a unit of measurement.

## ⚠ Caution

Changing the initial set of configurations has implications. Before proceeding with any change, familiarize yourself with the warnings in [Units of Measurement \[page 1632\]](#).

## Import Parameters

Parameter Name	Value Help
UNIT	Internal unit of measurement

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error <b>'X':</b> Save error

### ① Note

Class exception CX\_UOM\_ERROR is raised to check the integrity of the data import parameters.

### ↔ Sample Code

```
CLASS zcl_uom_unit_delete_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_unit_delete_test IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.
    DATA(lo_uom) = cl_uom_maintenance=>get_instance( ).

    TRY.
      lo_uom->delete( EXPORTING unit = 'ZYX'
                      IMPORTING error = DATA(lv_error) ).

      CATCH cx_uom_error INTO DATA(lo_error).
        out->write( | Exception raised | ).
        out->write( lo_error->get_text( ) ).

    ENDTRY.
    IF lv_error = abap_true.
      out->write( |Error occurred while deleting the data from the
database| ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.
```

## 4.2.11.12.2.4 Reading a Unit of Measurement

Use method `READ` to read a unit of measurement.

### Import Parameters

Parameter Name	Value Help
UNIT	Internal unit of measurement

### Export Parameters

Parameter Name	Field Name	Value Help
UNIT_ST		Structure for reading a unit of measurement
	UNIT	Internal unit of measurement
	COMMERCIAL	Commercial/external measurement unit format
	TECHNICAL	Technical measurement unit format
	DEC_ROUND	Number of decimal places to which this measurement unit should be rounded for conversion
	DIMID	Dimension key
	NUMERATOR	Numerator for conversion to SI unit
	DENOMINATOR	Denominator for conversion into SI unit
	EXPONENT	Base ten exponent for conversion to SI unit
	CONSTANT	Additive constant for conversion to SI unit
	DEC_DISP	Number of decimal places with which this measurement unit is displayed

Parameter Name	Field Name	Value Help
	ISOCODE	ISO code for measurement units. An ISO code can be assigned to several internal measurement units of a dimension.
	PRIMARY	Unit of measure flagged as a primary unit for an ISO code  <b>Space:</b> Not primary  <b>X:</b> Set as primary
	TEXT	Description of a unit of measurement
	LONG_TEXT	Long description of a unit of measurement

### ⓘ Note

Class exception CX\_UOM\_ERROR is raised if no unit is found.

### ↴ Sample Code

```

CLASS zcl_uom_unit_read_test DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_unit_read_test IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.

    DATA(lo_uom) = cl_uom_maintenance=>get_instance( ).
    TRY.
      lo_uom->read( EXPORTING unit = 'ST'
                     IMPORTING unit_st = DATA(ls_unit) ).

      CATCH cx_uom_error.
    ENDTRY.
    out->write( ls_unit-unit ).
    out->write( ls_unit-commercial ).
    out->write( ls_unit-technical ).
    out->write( ls_unit-dimid ).

  ENDMETHOD.
ENDCLASS.

```

## 4.2.11.12.3 Conversion Functions for Units of Measurement

Class CL\_UOM\_CONVERSION provides methods for simple conversion functions for units of measurement.

For more information, see the following:

- [Simple Conversion Between Two Units \[page 1649\]](#)
- [Determining the SI Unit \[page 1651\]](#)
- [Determining the Conversion Factors \[page 1653\]](#)
- [Converting Fraction to Fraction with a 5-Digit Numerator/Denominator \[page 1654\]](#)

### 4.2.11.12.3.1 Simple Conversion Between Two Units

Use method UNIT\_CONVERSION\_SIMPLE to convert values from one measurement unit to another and round the result to the number of decimal places maintained in the measurement unit table T006, if necessary.

Depending on the parameter ROUND\_SIGN, the rounding is up ('+') or down ('-'), commercial ('x'), or no rounding (SPACE).

#### ⓘ Note

Make sure that both units are maintained in the measurement unit table and are not dimensionless, but have both the **same** dimension.

### Import Parameters

Parameter Name	Value Help
INPUT	Input value
NO_TYPE_CHECK	Conversion factor type check  'x': No check  Space: Type check
ROUND_SIGN	Rounding flag  '+': Up  '-': Down  'x': Commercial
UNIT_IN	Unit of input value

## Export Parameters

Parameter Name	Value Help
ADD_CONST	Additive constant for conversion
DECIMALS	Number of decimal places for rounding
DENOMINATOR	Denominator for conversion
NUMERATOR	Numerator for conversion
OUTPUT	Output value

## Exceptions

Exception Name	Value Help
CONVERSION_NOT_FOUND	Conversion factor could not be determined
DIVISION_BY_ZERO	Division by zero caught
INPUT_INVALID	Input value is not a number
OUTPUT_INVALID	Output parameter is not a number
OVERFLOW	Field overflow
TYPE_INVALID	Output parameter is not a number
UNITS_MISSING	No units specified
UNIT_IN_NOT_FOUND	UNIT_IN is not maintained
UNIT_OUT_NOT_FOUND	UNIT_OUT is not maintained

### ↔ Sample Code

```
CLASS zcl_uom_conversion DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_conversion IMPLEMENTATION.

  METHOD if_oo_adt_classrun~main.
```

```

DATA: lv_input  TYPE p DECIMALS 8 VALUE '1.98678923',
      lv_result TYPE p DECIMALS 8.
DATA(lo_unit) = cl_uom_conversion->create( ).

lo_unit->unit_conversion_simple( EXPORTING input           =
lv_input
                                         round_sign          = 'X'
                                         unit_in             = 'KG'
                                         unit_out            = 'G'
                                     IMPORTING output           =
lv_result
                                     EXCEPTIONS conversion_not_found = 01
                                         division_by_zero     = 02
                                         input_invalid       = 03
                                         output_invalid      = 04
                                         overflow            = 05
                                         units_missing        = 06
                                         unit_in_not_found   = 07
                                         unit_out_not_found  = 08 ).

IF sy-subrc = 0.
    out->write( lv_result ).
ENDIF.
ENDMETHOD.
ENDCLASS.

```

### 4.2.11.12.3.2 Determining the SI Unit

Use method SI\_UNIT\_GET to determine the SI unit.

#### Import Parameters

Parameter Name	Value Help
DIMENSION	Dimension key
UNIT	Unit of measurement

#### Export Parameters

Parameter Name	Value Help
SI_UNIT	Base unit/SI unit of a dimension

## Exceptions

Exception Name	Value Help
DIMENSION_NOT_FOUND	Dimension is not defined
UNIT_NOT_FOUND	Unit of measurement is not maintained
SI_UNIT_NOT_FOUND	SI unit is not defined

### ↔ Sample Code

```
CLASS zcl_uom_conversion DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_conversion IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA(lo_unit) = cl_uom_conversion=>create( ).
    lo_unit->si_unit_get(
      EXPORTING
        dimension      = 'MASS'
        unit          = 'G'
      IMPORTING
        si_unit       = DATA(lv_si_unit)
    EXCEPTIONS
      dimension_not_found = 1
      unit_not_found     = 2
      si_unit_not_found = 3
      OTHERS             = 4 ).

    IF sy-subrc = 0.
      out->write( lv_si_unit ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.
```

### 4.2.11.12.3.3 Determining the Conversion Factors

Use method `UNIT_PARAMETERS_GET` to determine the data conversion factors of a unit.

#### Import Parameters

Parameter Name	Value Help
UNIT	Unit of measurement

#### Export Parameters

Parameter Name	Value Help
DECIMALS	Number of decimal places for rounding
DIMENSION	Dimension key
NUMERATOR	Numerator for conversion to SI unit
DENOMINATOR	Denominator for conversion to SI unit
EXONENT	Base ten exponent for conversion to SI unit
ADD_CONST	Additive constant for conversion to SI unit
DECAN	Number of decimal places for number display
FAMUNIT	Unit of measurement family

#### Exceptions

Exception Name	Value Help
UNIT_NOT_FOUND	Unit of measurement is not maintained

#### ↔ Sample Code

```
CLASS zcl_uom_conversion DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.

  PUBLIC SECTION.
```

```

INTERFACES if_oo_adt_classrun.
PROTECTED SECTION.
PRIVATE SECTION.
ENDCLASS.

CLASS zcl_uom_conversion IMPLEMENTATION.
METHOD if_oo_adt_classrun~main.
  DATA(lo_unit) = cl_uom_conversion=>create( ).
  lo_unit->unit_parameters_get(
    EXPORTING
      unit          = 'G'
    IMPORTING
      decimals     = DATA(lv_decimal)
      dimension    = DATA(lv_dimension)
      numerator   = DATA(lv_numerator)
      denominator = DATA(lv_denominator)
  EXCEPTIONS
    unit_not_found = 1
    OTHERS         = 2 .
  IF sy-subrc = 0.
    out->write( lv_dimension ).
    out->write( lv_decimal ).
    out->write( lv_numerator ).
    out->write( lv_denominator ).
  ENDIF.
ENDMETHOD.
ENDCLASS.

```

#### 4.2.11.12.3.4 Converting Fraction to Fraction with a 5-Digit Numerator/Denominator

Use method CONVERT\_TO\_FRACT5 to convert a fraction to a fraction with a maximum of 5-digit numerators/denominators.

If the numerator or denominator contain more than 5 digits, the result will be an approximation as close as possible.

#### Import Parameters

Parameter Name	Value Help
NOMIN	Nominator input
DENOMIN	Denominator input

## Export Parameters

Parameter Name	Value Help
NOMOUT	Nominator output (5-digit max.)
DENOMOUT	Denominator output (5-digit max.)

## Exceptions

Exception Name	Value Help
CONVERSION_OVERFLOW	Unit of measurement is not maintained.

### ↔ Sample Code

```
CLASS zcl_uom_convert_to_fract5 DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC.
  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_convert_to_fract5 IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.
    DATA: lv_nomin      TYPE f VALUE 999998,
          lv_denomin   TYPE f VALUE 999999.

    DATA(lo_unit) = cl_uom_conversion=>create( ).
    TRY.
      lo_unit->convert_to_fract5(
        EXPORTING
          nomin      = lv_nomin
          denomin   = lv_denomin
        IMPORTING
          nomout     = DATA(lv_nomout)
          denomout   = DATA(lv_denomout) ).

      CATCH cx_uom_error INTO DATA(lo_error).
        out->write( |Exception raised| ).
        out->write( lo_error->get_text( ) ) .
    ENDTRY.
    out->write( lv_nomout ).
    out->write( lv_denomout ) .
  ENDMETHOD.
ENDCLASS.
```

## 4.2.11.13 Identity and Access Management

Get an overview of the ABAP classes and methods you can use to access business roles and business users.

The following classes and methods allow you to read, create and modify business roles and business users.

These classes and methods can only be used if the authorizations are defined via standard IAM catalogs.

Business Roles

Class/Method	Description
CL_IAM_BUSINESS_ROLE_FACTORY	Factory class for IF_IAM_BUSINESS_ROLE_FACTORY
IF_IAM_BUSINESS_ROLE_FACTORY	Allows you to query and retrieve business role instances
IF_IAM_BUSINESS_ROLE	Allows you to read and modify attributes of business roles

### Query for Business Roles

#### Sample Code

```
* Get instance
  data(lo_br_factory) = cl_iam_business_role_factory->create_instance( ).
* Query for Business Roles
  lo_br_factory->query_business_roles(
    exporting
      ir_brole_id = value #( ( sign = 'I'
option = 'CP'
low = 'SAP_BR_QUALITY_ENGINEER_DWH' ) )
    importing
      et_result = data(lt_result) .
* get details of the Business Roles
  loop at lt_result assigning field-symbol(<fs_result>).
* Get Business Role instance
  lo_br_factory->get_business_role(
    exporting
      iv_uuid      = <fs_result>-uuid
    importing
      eo_business_role = data(lo_brole)
      et_return     = data(lt_return) .
* Business Role ID
  data(lv_id) = lo_brole->get_id( ).
* Business Role description
  data(lv_description) = lo_brole->get_description( ).
  out->write( |{ lv_id }: { lv_description }| ).
* Get Business Role Access restrictions
  lo_brole->get_access_restriction(
    importing
      ev_write = data(lv_w)
      ev_read  = data(lv_r)
      ev_f4    = data(lv_f4) .
    out->write( |Access-Restrictions: Write: { lv_w },
               Read: { lv_r }, F4: { lv_f4 }| ).
* Get assigned Business Catalogs
  data(lt_bc) = lo_brole->get_business_catalogs( ).
* Get assigned Business Users
  data(lo_user_factory) =
  cl_iam_business_user_factory->create_instance( ).
  data(lt_users) = lo_brole->get_users( ).
  loop at lt_users assigning field-symbol(<fs_user>).
```

```

        out->write( <fs_user> ).  

        lo_user_factory->get_business_user(  

            exporting  

                iv_user_id      = <fs_user>

```

## Create Business Role from Business Role Template

### ↔ Sample Code

```

data: lv_id type if_iam_business_role=>ty_id.  

data(lo_br_factory) = cl_iam_business_role_factory=>create_instance( ).  

* Create Business From Business Role Template  

    lv_id = 'ZZ_MY_BR_ADMINISTRATOR'.  

    lo_br_factory->create_brole_from_template(  

        exporting  

            iv_id          = lv_id  

            iv_template_id = 'SAP_BR_ADMINISTRATOR'  

        importing  

            eo_business_role = data(lo_brole)  

            et_return       = data(lt_return) ).  

* Set Business Role: Read = unrestricted, Write = Unrestricted, F4 =  

unrestricted  

    lo_brole->set_access_restriction(  

        exporting  

            iv_write     = if_iam_business_role=>co_access_restriction_code-  

unrestricted  

            iv_read      = if_iam_business_role=>co_access_restriction_code-  

unrestricted  

            iv_f4        = if_iam_business_role=>co_access_restriction_code-  

unrestricted  

        importing  

            et_return   = lt_return ).  

* Assign Business User to Business Role  

    lo_brole->add_user(  

        exporting  

            iv_user_id = 'CB0000000051'  

        importing  

            et_return  = lt_return ).  

    lo_brole->save(  

        importing  

            et_return  = lt_return ).

```

## Create Restricted Business Role

### ↔ Sample Code

```

data: ls_restriction type if_iam_business_role=>ty_restriction,  

        lv_id           type if_iam_business_role=>ty_id.  

data(lo_br_factory) = cl_iam_business_role_factory=>create_instance( ).  

* Create Business from Business Role Template  

    lv_id = 'ZZ_MY_BUSINESS_ROLE'.  

    lo_br_factory->create_business_role(  

        exporting  

            iv_id          = lv_id

```

```

importing
    eo_business_role = data(lo_brole)
    et_return        = data(lt_return) .
lo_brole->set_description( 'My Business Role' ).
lo_brole->set_long_text( 'This Business Role grant access to ....' ) .
* Add Business Catalog
    lo_brole->add_business_catalog(
        exporting
            iv_bu_catalog_id = 'ZZ_MY_BUSINESS_CATALOG'
        importing
            et_return        = lt_return ) .
* Set Business Role: Read = Unrestricted, Write = Restricted, F4 =
Unrestricted
    lo_brole->set_access_restriction(
        exporting
            iv_write  = if_iam_business_role=>co_access_restriction_code-
restricted
            iv_read   = if_iam_business_role=>co_access_restriction_code-
unrestricted
            iv_f4     = if_iam_business_role=>co_access_restriction_code-
unrestricted
        importing
            et_return = lt_return ) .
* Define restriction values for Write access
    ls_restriction-access_category_code =
if_iam_business_role=>co_access_category_code-write.
    ls_restriction-restriction_type = 'ZZ_SALES_AREA'.
    ls_restriction-restriction_field_value = value #(
        ( field_name = 'ZSALES_ORG'
            values = value #( ( low_value = 'A' )
                                ( low_value = 'B' ) ) )
        ( field_name = 'ZDIST_CHANNEL'
            values = value #( ( low_value = 'A*' high_value = 'F*' ) ) )
        ( field_name = 'ZDIVISION'
            values = value #( ( low_value = '*' ) ) ) .
lo_brole->create_restriction(
    exporting
        is_restriction = ls_restriction
    importing
        es_restriction = data(ls_restriction_set)
        et_return      = lt_return ) .
* Assign Business User to Business Role
    lo_brole->add_user(
        exporting
            iv_user_id = 'CB0000000051'
        importing
            et_return  = lt_return ) .
lo_brole->save(
    importing
        et_return = lt_return ) .

```

## Business Users

Class/Method	Description
CL_IAM_BUSINESS_USER_FACTORY	Factory class for IF_IAM_BUSINESS_USER_FACTORY
IF_IAM_BUSINESS_USER_FACTORY	Allows you to query and retrieve business user instances
IF_IAM_BUSINESS_USER	Allows you to read and modify attributes of business users

## Query for Business Users and Their Assigned Business Roles

### ↳ Sample Code

```
* Get Business user by UserID
  data(lo_bu_factory) = cl_iam_business_user_factory->create_instance( ).
  lo_bu_factory->get_business_user(
    exporting
      iv_user_id      = 'CB0000000051'
    importing
      eo_business_user = data(lo_buser)
      et_return        = data(lt_return) ).

* Get User properties
  data(lv_user) = lo_buser->get_user_name( ).

* get assigned Business Roles
  data(lo_br_factory) = cl_iam_business_role_factory->create_instance( ).
  data(lt_brole_id) = lo_buser->get_business_roles( ).
  loop at lt_brole_id assigning field-symbol(<fs_brole_id>).
    lo_br_factory->get_business_role_by_id(
      exporting
        iv_id          = <fs_brole_id>
      importing
        eo_business_role = data(lo_brole)
        et_return       = lt_return ).
    out->write( | ID: { lo_brole->get_id( ) }| ).
  endloop.
```

## Create Business User and Add Business Role

### ↳ Sample Code

```
data(lo_bu_factory) = cl_iam_business_user_factory->create_instance( ).

* Create Business user for Employee ID
  lo_bu_factory->create_business_user(
    exporting
      iv_bupa_id     = '0000000023'
      iv_user_name   = 'ZMY_BUSINESS_USER'
    importing
      eo_business_user = data(lo_buser)
      et_return        = data(lt_return) ).

* Set Date Format to MM-DD-YYYY (Gregorian Date)
  lo_buser->set_date_format( '3' ).

* Set logon language to English
  lo_buser->set_logon_language( 'E' ).

* Assign Business Role to the User
  lo_buser->add_business_roles(
    exporting
      it_business_role_id = value #( ( 'ZZ_MY_BUSINESS_ROLE' ) )
    importing
      et_return           = lt_return ).

* Save Business user
  lo_buser->save(
    importing
      et_return          = lt_return ).
```

## Lock Business User

### ↳ Sample Code

```
* get business user user name
  data(lo_bu_factory) = cl_iam_business_user_factory->create_instance( ).
  lo_bu_factory->get_business_user(
    exporting
      iv_user_name      = 'ZMY_BUSINESS_USER'
    importing
      eo_business_user = data(lo_buser)
      et_return         = data(lt_return) .
* Get lock status
  lo_buser->get_lock_status(
    importing
      ev_locked = data(lv_locked) .
* Lock user
  if lv_locked = abap_false.
  lo_buser->lock(
    importing
      et_return = lt_return .
endif.
```

## 4.2.11.14 Communication Management

Get an overview of the ABAP classes and methods you can use to access communication scenarios, communication systems, communication arrangements and communication users.

The following classes and methods allow you to read communication scenarios, communication systems, communication arrangements and communication users.

These classes and methods can only be used if the authorizations are defined via standard COM catalogs.

### Communication Scenarios

Class/Method	Description
CL_COM_SCENARIO_FACTORY	Factory class for IF_COM_SCENARIO_FACTORY
IF_COM_SCENARIO_FACTORY	Allows you to query and retrieve communication scenarios
IF_COM_SCENARIO	Allows you to read attributes of communication scenarios

### Communication Users

Class/Method	Description
CL_COM_USER_FACTORY	Factory class for IF_COM_USER_FACTORY
IF_COM_USER_FACTORY	Allows you to query and retrieve communication users
IF_COM_USER	Allows you to read attributes of communication users

### Communication Systems

Class/Method	Description
CL_COM_SYSTEM_FACTORY	Factory class for IF_COM_SYSTEM_FACTORY

Class/Method	Description
IF_COM_SYSTEM_FACTORY	Allows you to query and retrieve communication systems
IF_COM_SYSTEM	Allows you to read attributes of communication systems
Communication Arrangements	
Class/Method	Description
CL_COM_ARRANGEMENT_FACTORY	Factory class for IF_COM_ARRANGEMENT_FACTORY
IF_COM_ARRANGEMENT_FACTORY	Allows you to query and retrieve communication arrangements
IF_COM_ARRANGEMENT	Allows you to read attributes of communication arrangements

## 4.2.11.15 Proxy API for the Workflow Capability

With the proxy API, you can start workflows of the workflow capability within SAP Workflow Management out of your ABAP Environment.

The overall starting point is the class CL\_SWF\_CPWF\_API\_FACTORY\_A4C. Once you have the API object of interface IF\_SWF\_CPWF\_API, see the ABAP documentation on supported actions.

### Prerequisites

You've executed the integration steps in your ABAP environment. See [Integrating Workflow \[page 2919\]](#).

#### ↔ Sample Code

This coding sample shows how to start a workflow of the workflow capability.

```

TYPES: BEGIN OF ty_context,
        some_property TYPE string,
        END OF ty_context.
DATA(lo_cpwf_api) = cl_swf_cpwf_api_factory_a4c->get_api_instance( ).
DATA(ls_context) = VALUE ty_context(
    some_property = 'someValue'
).
DATA(lv_context_json) = lo_cpwf_api->get_start_context_from_data(
    iv_data = ls_context
).
DATA(lv_cpwf_handle) = lo_cpwf_api->start_workflow(
    iv_cp_workflow_def_id = 'SAP Workflow Management Workflow Definition ID'
    iv_context            = lv_context_json
    iv_retention_time     = 30
    iv_callback_class      = 'ZCL_SWF_CPWF_CALLBACK'
).

```

## ↳ Sample Code

This coding sample shows a completion callback that includes reading the context.

```
METHOD if_swf_cpwf_callback~workflow_instance_completed.  
  TYPES: BEGIN OF ty_context,  
         some_result TYPE string,  
         END OF ty_context.  
  TRY.  
    data(lo_cpwf_api) = cl_swf_cpwf_api_factory_a4c=>get_api_instance( ).  
    data(lv_context_json) = lo_cpwf_api->  
    >get_workflow_context( iv_cpwf_handle = iv_cpwf_handle ).  
    DATA ls_context TYPE ty_context.  
    lo_cpwf_api->get_context_data_from_json(  
      EXPORTING  
        iv_context = lv_context_json  
      IMPORTING  
        ev_data = ls_context  
    ).  
    CATCH cx_swf_cpwf_api INTO DATA(lx_exc).  
    RAISE EXCEPTION TYPE zcx_swf_cpwf_callback  
      EXPORTING  
        previous = lx_exc.  
  ENDTRY.  
  " your business coding  
ENDMETHOD.
```

## ↳ Sample Code

This coding sample shows the raising of an event towards SAP Business Technology Platform.

```
TYPES: BEGIN OF ty_example_context,  
       value TYPE int4,  
       END OF ty_example_context.  
CONSTANTS: lc_cp_workflow_def_id TYPE  
if_swf_cpwf_api=>cpwf_def_id      VALUE '<SAP Workflow Management Workflow  
Definition ID>',  
           lc_event_def_id      TYPE  
if_swf_cpwf_api=>cpwf_evt_def_id   VALUE '<SAP Workflow Management Event  
Definition ID>' .  
TRY.  
  " Get a Instance for the CPWF Integration API  
  DATA(lo_cpwf_api) = cl_swf_cpwf_api_factory_a4c=>get_api_instance( ).  
  " There should be a started workflow in order to raise an event  
  DATA(lv_cpwf_handle) = lo_cpwf_api->start_workflow(  
    EXPORTING  
      iv_cp_workflow_def_id = lc_cp_workflow_def_id  
      iv_retention_time     = 30  
    ).  
  COMMIT WORK.  
  " provide relevant event context data => in this case a meaningful  
integer  
  DATA(lv_event_context_data) = VALUE ty_example_context( value =  
4711 ).  
  DATA(lv_event_context) = lo_cpwf_api->get_context_from_data( iv_data  
= lv_event_context_data ).  
  
  " actually raise the event  
  lo_cpwf_api->raise_event(  
    EXPORTING  
      iv_cpwf_handle     = lv_cpwf_handle  
      iv_event_def_id   = lc_event_def_id  
      iv_event_context  = lv_event_context  
    ).  
  COMMIT WORK.
```

```
CATCH cx_swf_cpwf_api INTO DATA(lx_api).
  WRITE: 'Exception occurred: ' && lx_api->get_longtext( ).
ENDTRY.
```

## ABAP RESTful Application Programming Model (RAP)

### Early Numbering

In the RAP version of the Workflow Service API, workflows can be registered using the following command:

#### ↔ Sample Code

Example: EML Create

```
MODIFY ENTITIES OF i_cpwf_inst
  ENTITY CPWFInstance
  EXECUTE registerWorkflow
  FROM VALUE #( ( %key = '<a UUID which is mandatory for mass enabled
callers>' )
                %param-RetentionTime = '<retention time in days>'
                %param-CpWfDefId = '<SAP Workflow Service Definition
ID>'
                %param-CallbackClass = '<Callback class for final state
handling>'
                %param-Consumer = '<SAP Workflow Scenario ID>' ).
```

#### ⓘ Note

It's not allowed to call the `registerWorkflow` method multiple times with an identical set of parameters. This also applies to cases where the otherwise optional `%key` parameter stays empty. Therefore, if you want to use the `registerWorkflow` method in a mass-enabled context you must provide the unique `%key` parameter.

The parameters `RetentionTime`, `CpWfDefId`, and `Consumer` must be provided.

The workflow context can be provided using the following EML statement:

#### ↔ Sample Code

Example: EML Create

```
MODIFY ENTITIES OF i_cpwf_inst
  ENTITY CPWFInstance
  EXECUTE setPayload
  FROM VALUE #( ( %key-CpWfHandle = '<required key (UUID) of the CPWF
instance>' )
                %param-context = '<workflow context in JSON
format>' ) ).
```

Both commands should be part of a determination on save. During the save sequence, the workflow is started with the provided data.

### Late Numbering

As for early numbering, the `registerWorkflow` method can be called in a determination on save for late numbering, too.

The workflow context can be set even later in the `adjust_numbers` method of your business object using the `setPayload` method.

#### ↔ Sample Code

Example: EML Create

```
MODIFY ENTITIES OF i_cpwf_inst
  ENTITY CPWFInstance
  EXECUTE setPayload
  FROM VALUE #( ( %key-CpWfHandle = <required key (UUID) of the CPWF
instance>
                %param-context = <workflow context in JSON format> ) ).
```

During the save sequence, the workflow is started with the provided data. The instance is removed from the draft table and is entered into the active table.

Note, that in the `adjust_numbers` method you must fill the key of each line of the mapped table to make sure the save sequence is executed successfully.

Note also that if you have multiple workflows per instance of your business object you might need to store the assignments yourself in an internal table to be able to call the `set_payload` method during `adjust_numbers` method with the correct data.

#### ↔ Sample Code

This coding sample shows reading the workflow instance context.

```
TYPES: BEGIN OF ty_context,
      attr1 TYPE string,
      attr2 TYPE int4,
      attr3 TYPE string,
    END OF ty_context.

DATA(lo_cpwf_api) = cl_swf_cpwf_api_factory->get_api_instance( 'DEFAULT' ).
DATA(lo_cp_json_with_empty) = lo_cpwf_api->get_json_converter( ).
DATA(lo_wf_inst_ctxt) = lo_cpwf_api-
>if_swf_cpwf_capi~workflow_instance_context( ).

DATA ls_context TYPE ty_context.
lo_wf_inst_ctxt->get_instance_context( EXPORTING iv_cpwf_handle =
<cpwf_handle> io_cp_json = lo_cp_json IMPORTING data = ls_context ).
```

### Cancel Workflow Capability Instances

To register a running workflow capability instance for cancelation, use the `registerWorkflowCancel` action using the command:

#### ↔ Sample Code

Example: EML Cancel

```
MODIFY ENTITIES OF i_cpwf_inst
  ENTITY CPWFInstance
  EXECUTE registerWorkflowCancel
  FROM VALUE #( ( %key = '<a UUID which is mandatory for mass enabled
callers>' ) ).
```

Note, that this action should be part of the interaction phase.

During the save sequence, the workflow instance is canceled in a background unit..

## 4.2.11.16 XCO Library

The XCO ("Extension Components") library is a general-purpose development library for ABAP aimed at providing a highly efficient ABAP development experience. The Cloud Platform (CP) edition of the XCO library is specifically designed to support ABAP development scenarios in the new SAP BTP, ABAP environment.

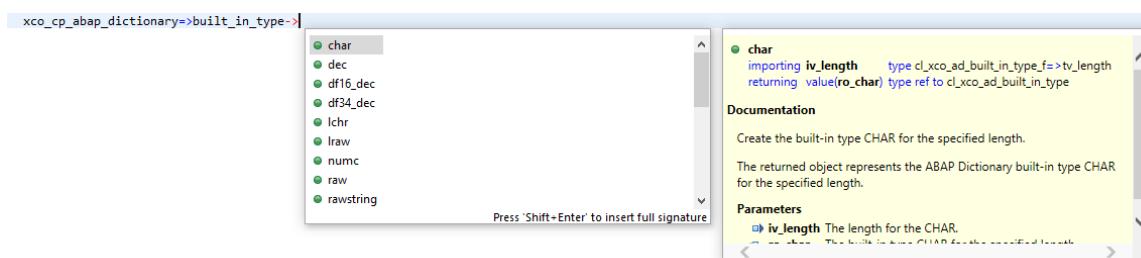
### 4.2.11.16.1 Core Principles

The design of the XCO library is based on the following core principles:

- **Modules**

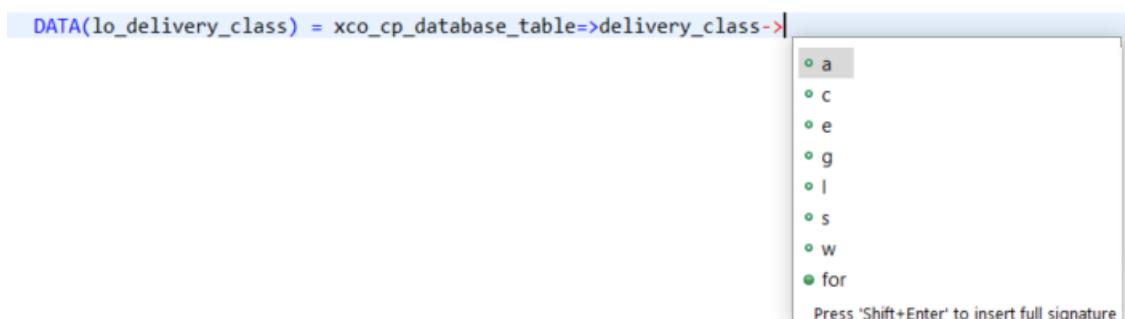
The XCO library is organized as a collection of (largely) independent modules (corresponding to ABAP packages). The public API of each module is exposed via a special class, called the API "class" of the module. It is distinguished from all other XCO classes and interfaces in that it starts with "XCO\_CP\_". The API class acts as the single point of entry for the functionality offered by a given module.

Starting with the API class, code completion can be used to easily discover the scope of a given module.



- **Enumerated Values**

In the XCO library enumerated values are represented by pre-created objects which can be accessed via the API class of the module they belong to. Besides offering a strongly typed way to work with fixed values they also provide additional properties specific to the enumeration as well as access to the underlying primitive value.



- **Exception Handling**

All XCO modules follow a uniform error propagation strategy. Unless the caller of an API can be expected to recover from an error situation (e.g. when attempting to acquire an exclusive lock), all error situations are handled through no\_check exceptions.

Any runtime exception that is raised explicitly by the XCO library is a subclass of CX\_XCO\_RUNTIME\_EXCEPTION (cf. section Exception philosophy in [Standard Library \[page 1790\]](#)).

## 4.2.11.16.2 Overview of XCO Modules

The Cloud Platform (CP) edition of the XCO library is comprised of the following modules:

- ABAP (XCO\_CP\_ABAP)  
Provides access to conceptual abstractions for classes and interfaces as well as to representations of ABAP built-in and generic types.
- Application log object (XCO\_CP\_APPLICATION\_LOG\_OBJECT)  
Provides access to text attributes specific to application log objects.
- ABAP Dictionary (XCO\_CP\_ABAP\_DICTIONARY)  
Provides access to conceptual abstractions for ABAP Dictionary elements (Database tables, data elements, structures, table types and domains) as well as to representations of ABAP Dictionary built-in types and their corresponding reference types.
- ABAP Objects (XCO\_CP\_ABAP\_OBJECTS)  
Provides access to enumerations specific to ABAP Objects, e.g. the visibility of class or interface components, as well as to supported origins for the XCO Read APIs for classes and interfaces.
- ABAP Repository (XCO\_CP\_ABAP\_REPOSITORY)  
Provides APIs and abstractions for retrieving and filtering development objects of the ABAP Repository in a strongly typed manner.
- ABAP SQL (XCO\_CP\_ABAP\_SQL)  
Provides ways to build ABAP SQL constraints to be used in conjunction with the filtering offerings of the XCO ABAP Repository module.
- ABAP Unit (XCO\_CP\_ABAP\_UNIT)  
Provides access to enumerations in the context of ABAP Unit.
- AMDP (XCO\_CP\_AMDP)  
Provides access to enumerations in the context of ABAP-managed database procedures.
- ARS (XCO\_CP\_ARS)  
Provides standard abstractions in the context of the API Release (ARS) framework (e.g. for compatibility contracts) to be used when programmatically setting or getting API states via the XCO ABAP Repository APIs.
- Business Application Log (XCO\_CP\_BAL)  
Provides APIs for creating, deleting and searching logs as well as standard abstractions for integrating logging functionality into custom application logic.
- Behavior definition (XCO\_CP\_BEHAVIOR\_DEFINITION)  
Provides access to enumerations specific to behavior definitions.
- Behavior implementation (XCO\_CP\_BEHAVIOR\_IMPLEMENTATION)  
Provides access to enumerations specific to behavior implementations.
- Binary (XCO\_CP\_BINARY)  
Provides access to binary text encodings, e.g. Base64.

- Business configuration object (XCO\_CP\_BUSINESS\_CNFGRTN\_OBJECT)  
Provides access to text attributes specific to business configuration objects.
- Call stack (XCO\_CP\_CALL\_STACK)  
Provides access to abstractions for programmatically working with all stacks (e.g. supported formats)
- Core Data Services (XCO\_CP\_CDS)  
Provides access to enumerations specific to the field of Core Data Services (CDS) as well as conceptual abstractions for behavior definitions, data definitions, metadata extensions and CDS entities.
- CDS Annotation (XCO\_CP\_CDS\_ANNOTATION)  
Provides ways to build CDS annotation values to be used when generating DDLS, DDLX or SRVD objects via the XCO Generation APIs.
- CDS Type Definition (XCO\_CP\_CDS\_TYPE\_DEFINITION)  
Provides access to enumerations specific to CDS type definitions as well as to supported versions and origins as they can be used in the context of the XCO Read APIs for CDS type definitions.
- Character (XCO\_CP\_CHARACTER)  
Provides access to abstractions useful when working with character-like data, e.g. code pages which can be used to translate between STRINGS and XSTRINGS
- Correction and Transport System (XCO\_CP\_CTS)  
Provides abstractions for working with the Correction and Transport System (CTS), e.g. for creating and releasing Workbench transport requests.
- Database table (XCO\_CP\_DATABASE\_TABLE)  
Provides access to database table specific enumerations, e.g. the size category of a database table, as well as to supported origins for the XCO Read APIs for database tables.
- Data definition (XCO\_CP\_DATA\_DEFINITION)  
Provides access to enumerations and text attributes specific to CDS data definitions.
- Data element (XCO\_CP\_DATA\_ELEMENT)  
Provides access to text attributes specific to data elements as well as to supported origins for the XCO Read APIs for data elements.
- Data control language (XCO\_CP\_DCL)  
Provides ways to build DCL (Data control language) expressions to be used when generating DCLS objects via the XCO Generation APIs
- Data definition language (XCO\_CP\_DDL)  
Provides ways to build DDL (Data definition language) expressions to be used when generating DDLS objects via the XCO Generation APIs.
- Domain (XCO\_CP\_DOMAIN)  
Provides access to text attributes specific to domains as well as to supported origins for the XCO Read APIs for domains.
- Event binding (XCO\_CP\_EVENT\_BINDING)  
Provides access to supported origins and versions for the XCO Read APIs for event bindings.
- Function group (XCO\_CP\_FUNCTION\_GROUP)  
Provides access to supported origins for the XCO Read APIs for function groups.
- Function module (XCO\_CP\_FUNCTION\_MODULE)  
Provides access to function module-specific enumerations as well as to supported origins for the XCO Read APIs for function modules.
- Generation (XCO\_CP\_GENERATION)  
Provides access to the XCO Generation APIs, i.e. allows to obtain a generation environment which can be used to create POST, PUT, PATCH and DELETE operations (as supported by the respective object type).
- Hash (XCO\_CP\_HASH)

Provides access to supported hash algorithms which can be used to calculate the hash value of strings with the XCO Standard Library.

- Internationalization (XCO\_CP\_I18N)  
Provides access to the XCO I18N APIs, i.e. allows to obtain domain, data element, data definition and message class targets which can be used to programmatically maintain language-dependent texts.
- Identify and Access Management (XCO\_CP\_IAM)  
Provides access to standard abstractions (e.g. for IAM business catalogs) within the area of Identify and Access Management.
- IAM Business Catalog  
Provides access to text attributes specific to IAM business catalogs.
- JSON (XCO\_CP\_JSON)  
Provides access to facilities used when working with JSON data in the context of the XCO standard library, such as the JSON builder or standard JSON transformations.
- Language (XCO\_CP\_LANGUAGE)  
Provides access to abstractions when working with languages, e.g. different formats.
- Message (XCO\_CP\_MESSAGE)  
Provides access to enumerations specific to messages, such as the message type.
- Message class (XCO\_CP\_MESSAGE\_CLASS)  
Provides access to text attributes specific to message classes.
- Metadata extension (XCO\_CP\_METADATA\_EXTENSION)  
Provides access to enumerations specific to metadata extensions.
- Name (XCO\_CP\_NAME)  
Provides access to IF\_XCO\_NAME\_CHOICES to be used in conjunction with POST operations in the context of the XCO Generation APIs.
- Package (XCO\_CP\_PACKAGE)  
Provides access to enumerations specific to packages.
- RESTful Application Programming (XCO\_CP\_RAP)  
Provides access to abstractions to integrate RAP specific types (e.g. RAP behavior messages) with standard abstractions (e.g. for business application logs) of the XCO Library.
- Regular expression (XCO\_CP\_REGULAR\_EXPRESSION)  
Provides access to abstractions used when working with regular expressions in the context of the XCO standard library, such as different regular expression engines.
- SAP object node type (XCO\_CP\_SAP\_OBJECT\_NODE\_TYPE)  
Provides access to enumerations specific to SAP object node types as well as to supported versions and origins as they can be used in the context of the XCO Read APIs for SAP object node types.
- SAP object type (XCO\_CP\_SAP\_OBJECT\_TYPE)  
Provides access to enumerations specific to SAP object types as well as to supported versions and origins as they can be used in the context of the XCO Read APIs for SAP object types.
- Service binding (XCO\_CP\_SERVICE\_BINDING)  
Provides access to enumerations specific to service bindings as well as to supported versions and origins that can be used in the context of XCO Read APIs for service bindings.
- Service consumption model (XCO\_CP\_SERVICE\_CONS\_MODEL)  
Provides access to supported versions and origins that can be used in the context of XCO Read APIs for service consumption models.
- Service definition (XCO\_CP\_SERVICE\_DEFINITION)  
Provides access to supported versions and origins as they can be used in the context of the XCO Read APIs for service definitions.

- Software component (XCO\_CP\_SOFTWARE\_COMPONENT)  
Provides access to enumerations specific to software components.
- String (XCO\_CP\_STRING)  
Provides access to abstractions used when working with strings in the context of the XCO standard library, such as the string builder or standard string compositions and decompositions.
- System (XCO\_CP\_SYSTEM)  
Provides access to abstractions for system-wide entities such as software components or application components..
- Table (XCO\_CP\_TABLE)  
Provides access to enumerations specific to tables (i.e. structures and database tables).
- Table type (XCO\_CP\_TABLE\_TYPE)  
Provides access to enumerations specific to table types.
- Tenant (XCO\_CP\_TENANT)  
Provides access to enumerations specific to tenants.
- Time (XCO\_CP\_TIME)  
Provides access to the XCO time library.
- Transport (XCO\_CP\_TRANSPORT)  
Provides access to enumerations and abstractions specific to transports.
- UUID (XCO\_CP\_UUID)  
Provides access to abstractions used when working with UUIDs in the context of the XCO standard library, such as different UUID formats.
- XSLX (XCO\_CP\_XSLX, XCO\_CP\_XLSX\_SELECTION, XCO\_CP\_XLSX\_READ\_ACCESS, XCO\_CP\_XLSX\_WRITE\_ACCESS  
)  
The XCO XLSX module supports the programmatic processing of XLSX documents. The various external APIs provide access to central abstractions required when reading data from XLSX documents.

## 4.2.11.16.3 ABAP Repository

### 4.2.11.16.3.1 Query APIs

The XCO ABAP Repository module provides Query APIs that allow to easily navigate through the ABAP Repository.

#### Context

To this extent, an object selection functionality is provided that is based on the following two central abstractions:

- Object source: An entity that can naturally act as a source of ABAP repository objects, e.g. a package or a transport or even the complete ABAP repository
- Object filter: An encapsulation for any kind of filter that can be applied to a selection of ABAP Repository objects. Examples are software or application components or type specific properties like foreign key attributes of database table fields

The entry point for retrieving a collection of objects from the ABAP Repository is XCO\_CP\_ABAP\_REPOSITORY=>OBJECTS. Note that independently of any filters, the visible ABAP Repository objects are those which either have been explicitly released by SAP or which belong to customer software components. The overall usage is illustrated by two examples below.

In the first example, all repository objects of the whole ABAP Repository whose software component is ZLOCAL and whose name contains the fragment CAL are retrieved. As no explicit object type is specified the retrieved objects have the type IF\_XCO\_AR\_OBJECT:

#### ↔ Sample Code

```
DATA(lo_software_component_filter) = xco_cp_system->software_component-
>get_filter( xco_cp_abap_sql=>constraint->equal( 'ZLOCAL' ) ).
DATA(lo_name_filter) = xco_cp_abap_repository->object_name-
>get_filter( xco_cp_abap_sql=>constraint->contains_pattern( '%CAL%' ) ).
DATA(lt_objects) = xco_cp_abap_repository->objects->where( VALUE #(
    ( lo_software_component_filter )
    ( lo_name_filter )
) )->in( xco_cp_abap=>repository )->get( ).
LOOP AT lt_objects INTO DATA(lo_object).
    DATA(lv_object_type) = lo_object->type->value.
    DATA(lv_object_name) = lo_object->name->value.
    " ...
ENDLOOP.
```

In the second example all database tables of the package Z\_MY\_OBJECTS that have at least one field which has a foreign key whose check table is ZHOLIDAY are retrieved. As in this case the object type (database table) is explicitly specified the retrieved objects are of type IF\_XCO\_DATABASE\_TABLE:

#### ↔ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( 'Z_MY_OBJECTS' ) .
DATA(lo_filter) = xco_cp_table->field_property->foreign_key_check_table-
>get_filter(
    xco_cp_abap_sql=>constraint->equal( 'ZHOLIDAY' )
).

DATA(lt_database_tables) = xco_cp_abap_repository->objects->tabl-
>database_tables->where( VALUE #(
    ( lo_filter )
) )->in( lo_package )->get( ).

LOOP AT lt_database_tables INTO DATA(lo_database_table).
    DATA(lv_name) = lo_database_table->name.

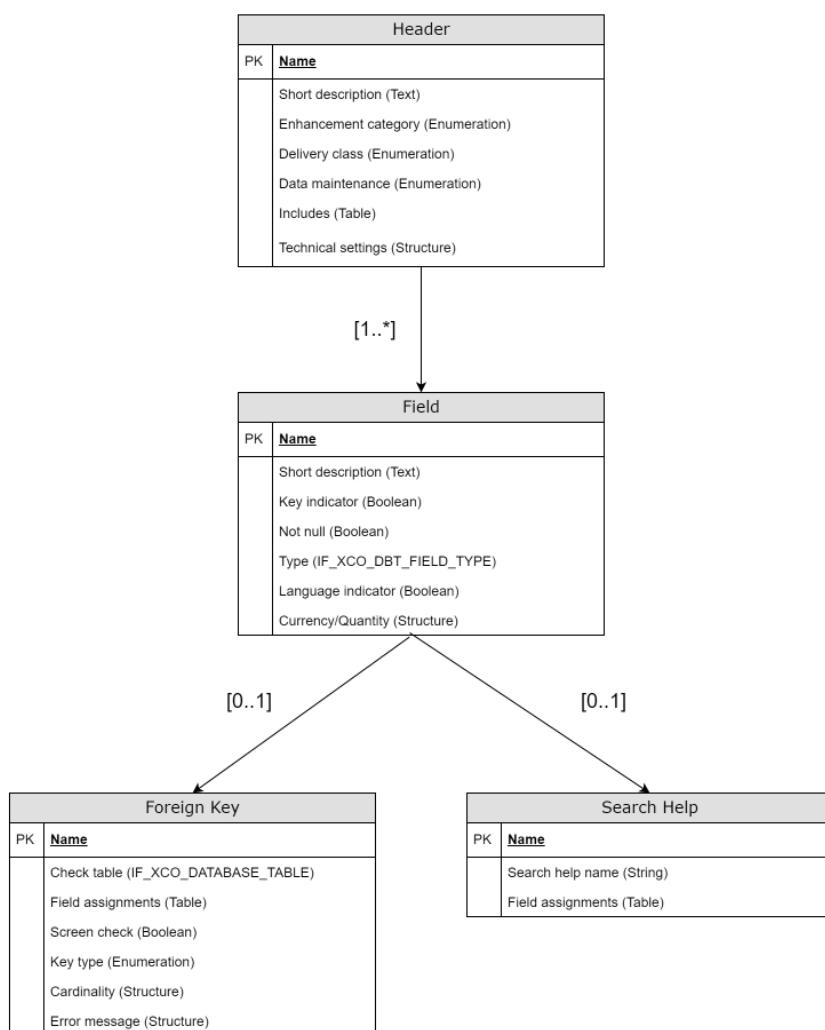
    " ...
ENDLOOP.
```

## 4.2.11.16.3.2 Read APIs

The XCO Read APIs provide homogenous, yet strongly-typed, APIs that allow to access the content of ABAP Repository objects in a structured way.

### Hierarchical Object Model

The overall design of the XCO Read APIs (regardless of the concrete object type) is based on a natural hierarchical model for each respective object type. The hierarchical model used by the XCO Read APIs regards an object as a rooted tree, with the root node given by the header of the object and all other nodes corresponding to parts of the object. To illustrate, the diagram below contains the hierarchical object model for a database table of the ABAP Dictionary as used by the XCO Read APIs:



For each node in the hierarchical object model for an object type, two abstractions are provided:

- Object/part handle
- Content handle

## Object/Part Handles

The object/part handle provides an abstraction which encapsulates all key values (object name and keys of the parts if applicable) of an object/part. For the database table example above we have:

- IF\_XCO\_DATABASE\_TABLE: The object handle for the database table (encapsulating the name of the database table)
- IF\_XCO\_DBT\_FIELD: The handle for the field of a database table (encapsulating the name of the database table and the name of the field)
- IF\_XCO\_DBT\_FOREIGN\_KEY: The handle for the foreign key of a field of a database table (encapsulating the name of the database table and the name of the field for which to consider the foreign key)
- IF\_XCO\_DBT\_SEARCH\_HELP: The handle for the search help of a field of a database table (encapsulating the name of the database table and the name of the field for which to consider the search help)

The following code sample illustrates how these four handles can be obtained:

### ↔ Sample Code

```
" Handle for database table ZMY_DBT (IF_XCO_DATABASE_TABLE).
DATA(lo_database_table) = xco_cp_abap_dictionary=>database_table( 'ZMY_DBT' ).

" Handle for the field KEY_FIELD of database table ZMY_DBT (IF_XCO_DBT_FIELD).
DATA(lo_field) = lo_database_table->field( 'KEY_FIELD' ).

" Handle for the foreign key of field KEY_FIELD of database table ZMY_DBT
(IF_XCO_DBT_FOREIGN_KEY).
DATA(lo_foreign_key) = lo_field->foreign_key.

" Handle for the search help of field KEY_FIELD of database table ZMY_DBT
(IF_XCO_DBT_SEARCH_HELP).
DATA(lo_search_help) = lo_field->search_help.
```

A given object/part handle allows to check the existence via the EXISTS method and provides access to the content handle of the respective object/part via the CONTENT method.

## Content Handles

Content handles exist in a natural 1:1 relationship with object/part handles and can be obtained via the CONTENT method of the object/part handle. In case the object type supports different versions (e.g. active and inactive) a choice for the version needs to be made before obtaining a content handle. More generally, a content handle fixes all degrees of freedom except for the origin (see below) such that the attributes exposed by the content handle can be read out.

A given content handle provides a dedicated GET\_\* method for each exposed attribute as well as a method GET which will return all exposed attributes in a single structure. Note that for language-dependent attributes, e.g. short descriptions, the value will always be retrieved in the original language of the underlying ABAP Repository object.

For the database table example we have:

- IF\_XCO\_DBT\_CONTENT: The content handle for the header of the database table
- IF\_XCO\_DBT\_FIELD\_CONTENT: The content handle for a field of a database table

- IF\_XCO\_DBT\_FOREIGN\_KEY\_CONTENT: The content handle for the foreign key of a field of a database table
- IF\_XCO\_DBT\_SEARCH\_HELP\_CONTENT: The content handle for the search help of a field of a database table

The code sample below illustrates how the content handle for a database table field is obtained and used to read out the attributes of the field:

### ↔ Sample Code

```
" Content handle for the field KEY_FIELD of database table ZMY_DBT.
DATA(lo_field_content) = xco_cp_abap_dictionary=>database_table( 'ZMY_DBT'
    )->field( 'KEY_FIELD'
    )->content( ).

" Get the complete content (i.e. all the available attributes of the field)
" as a structure of all attributes.
DATA(ls_field_content) = lo_field_content->get( ).

" Get each attribute individually.
DATA(lv_short_description) = lo_field_content->get_short_description( ).
DATA(lv_key_indicator) = lo_field_content->get_key_indicator( ).
DATA(lv_not_null) = lo_field_content->get_not_null( ).
DATA(lo_type) = lo_field_content->get_type( ).
DATA(lv_language_indicator) = lo_field_content->get_language_indicator( ).
DATA(ls_currency_quantity) = lo_field_content->get_currency_quantity( ).
```

## Origins

An origin is an abstraction that encapsulates where the content (and existence status) of an object or part thereof are read from. As explained above, object/part as well as content handles are only backed by the semantic identifiers (e.g. the name of the database table) but do not make any choices if the content is read from the local system or from elsewhere. To this extent, each GET\* method on a content handle asks for an (optional) origin from which the attribute will henceforth be read. In short:

- Content handle: Encapsulates **what** shall be read
- Origin: Encapsulates **where** content shall be read from

As of now, the XCO Read APIs define the following three kinds of origins (availability depending on the implementation status of each object type):

- Default: The default origin will be used when no explicit origin is supplied. With the default origin, an object will be read exactly once per ABAP session from the local system. Once read from the local database, the content will be cached for the duration of the ABAP Session and used for any subsequent reads
- Local: The local origin reads from the local database. The caching behavior for individual or groups of objects can be configured freely
- Remote: Based on an RFC destination (given as IF\_RFC\_DEST) objects are read from remote systems. The caching behavior for individual or groups of objects can be configured freely

Note that not all object types support the origin concept yet (see table below). In case an object type does not yet support the origin concept the content will always be read from the local database.

## Object Type Feature Matrix

Object Type		Supports Origins?	Origin Factory	Supports Local Origin	Supports Remote Origin
APLO	Application Log Object	No	-	-	-
CLAS	Class	Yes	XCO_CP_ABAP_OBJECTS=>ORIGIN	Yes	Yes
DDLS	Data Definition	No	-	-	-
DOMA	Domain	Yes	XCO_CP_DOMAIN=>ORIGIN	Yes	No
DRTY	CDS Type Definition	Yes	XCO_CP_CDS_TYPE_DEFINITION	Yes	No
DTEL	Data Element	Yes	XCO_CP_DATA_ELEMENT=>ORIGIN	Yes	No
EVTB	Event Binding	Yes	XCO_CP_EVENT_BINDING=>ORIGIN	Yes	No
FUGR	Function Group	Yes	XCO_CP_FUNCTION_GROUP=>ORIGIN	Yes	No
	Function Module	Yes	XCO_CP_FUNCTION_MODULE=>ORIGIN	Yes	No
INTF	Interface	Yes	XCO_CP_ABAP_OBJECTS=>ORIGIN	Yes	Yes
MSAG	Message Class	No	-	-	-
NONT	SAP Object Node Type	Yes	XCO_CP_SAP_OBJECT_NODE_TYPE=>ORIGIN	Yes	No
RONT	SAP Object Type	Yes	XCO_CP_SAP_OBJECT_TYPE=>ORIGIN	Yes	No
SRVB	Service Binding	Yes	XCO_CP_SERVICE_BINDING=>ORIGIN	Yes	No
SRVC	Service Consumption Model	Yes	XCO_CP_SERVICE_CONSUMPTION_MODEL=>ORIGIN	Yes	No
SRVD	Service Definition	Yes	XCO_CP_SERVICE_DEFINITION=>ORIGIN	Yes	No
TABL	Structure	No	-	-	-

Object Type	Supports Origins?	Origin Factory	Supports Local Origin	Supports Remote Origin
Database Table	Yes	XCO_CP_DA-TABASE_TA-BLE=>ORIGIN	Yes	No
Global Temporary Table	No	-	-	-
TTYP	Table Type	No	-	-
XSLT	Transformation	No	-	-

### **4.2.11.16.3.2.1 Reading a Class**

Given the following class

```

CLASS zcl_xco_doc_cp_cs_class DEFINITION PUBLIC FINAL CREATE PUBLIC.
  PUBLIC SECTION.
    CLASS-METHODS:
      " ! <p class="shorttext synchronized" lang="en">
      " !   Calculate the factorial for the given natural number
      " !
      " !
      " ! @parameter iv_n |
      " !   <p class="shorttext synchronized" lang="en">
      " !     The natural number for which to calculate the factorial
      " !
      " ! @parameter rv_factorial_n |
      " !   <p class="shorttext synchronized" lang="en">
      " !     The factorial for the given natural number
      " !
      " ! @raising cx_abap_invalid_value |
      " !   <p class="shorttext synchronized" lang="en">
      " !     If IV_N is less than 0
      " !
      factorial
        IMPORTING
          iv_n                      TYPE i
        RETURNING
          VALUE(rv_factorial_n) TYPE i
        RAISING
          cx_abap_invalid_value.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_class IMPLEMENTATION.
  METHOD factorial.
    IF iv_n LT 0.
      RAISE EXCEPTION TYPE cx_abap_invalid_value.
    ELSEIF iv_n EQ 0.
      rv_factorial_n = 1.
    ELSE.
      rv_factorial_n = iv_n * factorial( iv_n - 1 ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.

```

its contents can be read and written to the console as follows

## ↔ Sample Code

```
CLASS zcl_xco_doc_cp_cs_read_clas DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
  constructor.
PROTECTED SECTION.
METHODS:
  main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:
  co_class_name  TYPE sxco_ao_object_name VALUE 'ZCL_XCO_DOC_CP_CS_CLASS'.
METHODS:
  write_parameters
    IMPORTING
      io_method TYPE REF TO if_xco_ao_c_method
      out       TYPE REF TO if_xco_adt_classrun_out,
  write_exceptions
    IMPORTING
      io_method TYPE REF TO if_xco_ao_c_method
      out       TYPE REF TO if_xco_adt_classrun_out,
  write_source
    IMPORTING
      io_class      TYPE REF TO if_xco_ao_class
      iv_method_name TYPE sxco_clas_method_name
      out          TYPE REF TO if_xco_adt_classrun_out.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_read_clas IMPLEMENTATION.
METHOD constructor.
  super->constructor( ).
ENDMETHOD.

METHOD main.
  DATA(lo_class) = xco_cp_abap=>class( co_class_name ).
  " DEFINITION.
  DATA(ls_definition_content) = lo_class->definition->content( )->get( ).
  out->write( |Definition of { lo_class->name }:| ).
  out->write( ls_definition_content ).
  " PUBLIC class methods.
  DATA(lt_public_methods) = lo_class->definition->section-public-
>components->class_method->all->get( ).
  LOOP AT lt_public_methods INTO DATA(lo_public_method).
    out->write( |`nClass method { lo_public_method->name }:| ).
    DATA(ls_public_method) = lo_public_method->content( )->get( ).
    out->write( ls_public_method ).
    out->write( |`nParameters:| ).
    write_parameters(
      io_method = lo_public_method
      out       = out
    ).
    out->write( |`nExceptions:| ).
    write_exceptions(
      io_method = lo_public_method
      out       = out
    ).
    out->write( |`nSource:| ).
    write_source(
      io_class      = lo_class
      iv_method_name = lo_public_method->name
      out          = out
    ).
  ENDLOOP.
ENDMETHOD.

METHOD write_parameters.
  " IMPORTING parameters.
  DATA(lt_importing_parameters) = io_method->importing_parameters->all-
>get( ).
```

```

LOOP AT lt_importing_parameters INTO DATA(lo_importing_parameter).
  out->write( lo_importing_parameter->name ).
  DATA(ls_importing_parameter) = lo_importing_parameter->content( )-
>get( ).
  out->write( ls_importing_parameter ).
ENDLOOP.
" RETURNING parameters.
DATA(lt_returning_parameters) = io_method->returning_parameters->all-
>get( ).
  LOOP AT lt_returning_parameters INTO DATA(lo_returning_parameter).
    out->write( lo_returning_parameter->name ).
    DATA(ls_returning_parameter) = lo_returning_parameter->content( )-
>get( ).
    out->write( ls_returning_parameter ).
  ENDLOOP.
ENDMETHOD.

METHOD write_exceptions.
  DATA(lt_exceptions) = io_method->exceptions->all->get( ).
  LOOP AT lt_exceptions INTO DATA(lo_exception).
    out->write( lo_exception->name ).
    DATA(ls_exception) = lo_exception->content( )->get( ).
    out->write( ls_exception ).
  ENDLOOP.
ENDMETHOD.

METHOD write_source.
  DATA(ls_implementation) = io_class->implementation->method( iv_method_name
  )->content( )->get( ).
  DATA(lv_source) = xco_cp=>strings( ls_implementation-source
  )->join( |{ cl_abap_char_utilities=>cr_lf }|
  )->value.
  out->write( lv_source ).
ENDMETHOD.

ENDCLASS.

```

## 4.2.11.16.3.2.2 Reading AMDP Attributes of a Method Implementation

Given a class with an AMDP method implementation such as

### ↳ Sample Code

```

CLASS zcl_xco_doc_cp_amdp DEFINITION PUBLIC FINAL CREATE PUBLIC.
  PUBLIC SECTION.
    INTERFACES:
      if_amdp_marker_hdb.

    METHODS:
      amdp_method.
  ENDCLASS.

CLASS zcl_xco_doc_cp_amdp IMPLEMENTATION.
  METHOD amdp_method BY DATABASE PROCEDURE FOR HDB LANGUAGE SQLSCRIPT
    OPTIONS
      READ-ONLY
      USING
        zxco_amdp_dbt.
    SELECT * FROM zxco_amdp_dbt WHERE id = '001';
  ENDMETHOD.

ENDCLASS.

```

the AMDP attributes of the method implementation can be read out as follows:

#### ↳ Sample Code

```
"! Code sample for reading the AMDP attributes of a method implementation.
"!
"! IMPORTANT: The values of the constants CO_CLASS_NAME and CO_METHOD_NAME
"! must be replaced with the name of an existing class and a method that is
"! implemented in this class.
CLASS zcl_xco_doc_cp_read_amdp DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PROTECTED SECTION.
METHODS:
    main REDEFINITION.

PRIVATE SECTION.
CONSTANTS:
    co_class_name  TYPE sxco_ao_object_name VALUE 'ZCL_XCO_DOC_CP_AMDP',
    co_method_name TYPE sxco_clas_method_name VALUE 'AMDP_METHOD'.
ENDCLASS.

CLASS zcl_xco_doc_cp_read_amdp IMPLEMENTATION.
METHOD main.
    DATA(lo_methodImplementation) = xco_cp_abap=>class( co_class_name
        )->implementation->method( co_method_name ).

    " LS_METHOD_IMPLEMENTATION will contain both the AMDP attributes and the
    source
    " code of the method implementation.
    DATA(ls_methodImplementation) = lo_methodImplementation->content( )-
>get( ).
    out->write( ls_methodImplementation ).

    " The AMDP database type is represented as an object of type
    CL_XCO_AMDP_DB_TYPE.
    " Supported AMDP database types are available via
    XCO_CP_AMDP=>DATABASE_TYPE.
    DATA(lo_amdp_database_type) = ls_methodImplementation-amdp-database_type.
    out->write( lo_amdp_database_type ).

    " The AMDP database language is represented as an object of type
    CL_XCO_AMDP_DB_LANGUAGE.
    " Supported AMDP database languages are available via
    XCO_CP_AMDP=>DATABASE_LANGUAGE.
    DATA(lo_amdp_database_language) = ls_methodImplementation-amdp-
database_language.
    out->write( lo_amdp_database_language ).

    " The AMDP database options are represented as objects of type
    CL_XCO_AMDP_DB_OPTION.
    " Supported AMDP database options are available via
    XCO_CP_AMDP=>DATABASE_OPTION.
    DATA(lt_amdp_database_options) = ls_methodImplementation-amdp-
database_options.
    out->write( lt_amdp_database_options ).

    " An AMDP database entity is represented as object of type
    IF_XCO_AMDP_DATABASE_ENTITY.
    " The IF_XCO_PRINTABLE~GET_TEXT method can be used to obtain the string
    value for a database
    " entity.
    LOOP AT ls_methodImplementation-amdp-database_entities INTO
    DATA(lo_amdp_database_entity).
        DATA(lv_amdp_database_entity) = lo_amdp_database_entity-
>if_xco_printable~get_text(
            )->get_lines(
            )->join(
```

```

        )->value.
        out->write( lv_amdp_database_entity ).
    ENDLOOP.
ENDMETHOD.
ENDCLASS.

```

### 4.2.11.16.3.2.3 Reading a Function Module

Given the following function module

#### ↳ Sample Code

```

FUNCTION z_xco_doc_cp_cs_fm
IMPORTING
  VALUE(im_class) TYPE REF TO cl_xco_ad_built_in_type OPTIONAL
  im_interface TYPE REF TO if_xco_news
  VALUE(im_abap_builtin_type) TYPE i DEFAULT 5
  VALUE(im_abap_gt_reference) TYPE REF TO data
EXPORTING
  ex_p1 TYPE REF TO string
  VALUE(ex_p2) TYPE c
CHANGING
  ch_1 TYPE any
  VALUE(ch_2) TYPE i
RAISING
  cx_sy_zerodivide
  RESUMABLE(cx_sy_assign_cast_error).

IF im_class IS NOT INITIAL.
  ch_1 = 'ABC'.
ENDIF.

ENDFUNCTION.

```

its contents can be read and written to the console as follows

#### ↳ Sample Code

```

CLASS zcl_xco_doc_cp_cs_fnctn_mdle DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PROTECTED SECTION.
  METHODS:
    main REDEFINITION.

  PRIVATE SECTION.
  METHODS:
    read_importing_parameters
      IMPORTING
        io_function_module TYPE REF TO if_xco_function_module
        out          TYPE REF TO if_xco_adt_classrun_out,

    read_exporting_parameters
      IMPORTING
        io_function_module TYPE REF TO if_xco_function_module
        out          TYPE REF TO if_xco_adt_classrun_out,

    read_changing_parameters

```

```

IMPORTING
    io_function_module TYPE REF TO if_xco_function_module
    out           TYPE REF TO if_xco_adt_classrun_out,
    read_exceptions
IMPORTING
    io_function_module TYPE REF TO if_xco_function_module
    out           TYPE REF TO if_xco_adt_classrun_out.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_fnctn_mdle IMPLEMENTATION.
METHOD main.
    DATA(lo_function_module) =
xco_cp_abap->function_module( 'Z_XCO_DOC_CP_CS_FM' ) .
    out->write( lo_function_module ).

    DATA(ls_function_module) = lo_function_module->content( )->get( ).
    out->write( ls_function_module ).

    out->write( | \nIMPORTING parameters:| ) ##NO_TEXT.
    read_importing_parameters(
        io_function_module = lo_function_module
        out           = out
    ).

    out->write( | \nEXPORTING parameters:| ) ##NO_TEXT.
    read_exporting_parameters(
        io_function_module = lo_function_module
        out           = out
    ).

    out->write( | \nCHANGING parameters:| ) ##NO_TEXT.
    read_changing_parameters(
        io_function_module = lo_function_module
        out           = out
    ).

    out->write( | \nExceptions:| ) ##NO_TEXT.
    read_exceptions(
        io_function_module = lo_function_module
        out           = out
    ).
ENDMETHOD.

METHOD read_importing_parameters.
    DATA(lt_importing_parameters) = io_function_module->importing_parameters-
>all->get( ).

    LOOP AT lt_importing_parameters INTO DATA(lo_importing_parameter).
        out->write( lo_importing_parameter ).

        DATA(ls_type) = lo_importing_parameter->content( )->get( ).
        out->write( ls_type ).
    ENDLOOP.
ENDMETHOD.

METHOD read_exporting_parameters.
    DATA(lt_exporting_parameters) = io_function_module->exporting_parameters-
>all->get( ).

    LOOP AT lt_exporting_parameters INTO DATA(lo_exporting_parameter).
        out->write( lo_exporting_parameter ).

        DATA(ls_type) = lo_exporting_parameter->content( )->get( ).
        out->write( ls_type ).
    ENDLOOP.
ENDMETHOD.

```

```

METHOD read_changing_parameters.
  DATA(lt_changing_parameters) = io_function_module->changing_parameters-
>all->get( ).

  LOOP AT lt_changing_parameters INTO DATA(lo_changing_parameter).
    out->write( lo_changing_parameter ).

    DATA(ls_type) = lo_changing_parameter->content( )->get( ).

    out->write( ls_type ).
  ENDLOOP.
ENDMETHOD.

METHOD read_exceptions.
  DATA(lt_exceptions) = io_function_module->exceptions->all->get( ).

  LOOP AT lt_exceptions INTO DATA(lo_exceptions).
    out->write( lo_exceptions ).

    DATA(ls_type) = lo_exceptions->content( )->get( ).

    out->write( ls_type ).
  ENDLOOP.
ENDMETHOD.

ENDCLASS.

```

#### 4.2.11.16.3.2.4 Reading a Transformation

Given a transformation (i.e. an XSLT object), its contents can be read and written to the console as follows:

##### ↔ Sample Code

```

"! Code sample for reading the contents of a transformation (XSLT object).
"!
"! IMPORTANT: The value of the constant CO_TRANSFORMATION_NAME must be
replaced with
"! the name of an existing transformation for which the content shall be read.
CLASS zcl_xco_doc_cp_read_trnsfrmtn DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PROTECTED SECTION.
  METHODS:
    main REDEFINITION.

  PRIVATE SECTION.
  CONSTANTS:
    co_transformation_name TYPE sxco_tf_object_name VALUE
    'ZMY_TRANSFORMATION'.
ENDCLASS.

CLASS zcl_xco_doc_cp_read_trnsfrmtn IMPLEMENTATION.
  METHOD main.
    DATA(lo_transformation) = xco_cp_abap_repository=>object->xslt-
>for( co_transformation_name ).
    out->write( lo_transformation ).

    " LS_TRANSFORMATION will contain both the short description and the
    source code of the
    " transformation.
    DATA(ls_transformation) = lo_transformation->content( )->get( ).

    out->write( ls_transformation ).

    " The source code of the transformation is represented as an object of
    type IF_XCO_TF_SOURCE_CODE.

```

```

    " Via the IF_XCO_TEXT~GET_LINES method the source code can also be easily
 obtained as a string_table.
    DATA(lt_source_code) = ls_transformation-source_code-
>if_xco_text~get_lines( )->value.
    out->write( lt_source_code ).
ENDMETHOD.
ENDCLASS.
```

## 4.2.11.16.3.2.5 Reading Classes and Interfaces from a Remote Destination

Besides the default and local origin, the Read APIs for both classes and interfaces offer the option to read the content of classes and interfaces from remote systems. Remote systems are identified via an IF\_RFC\_DEST for which objects can be obtained via CL RFC\_DESTINATION\_PROVIDER. Origins for classes and interfaces are obtained via XCO\_CP\_ABAP\_OBJECTS=>ORIGIN which provides a common basis for both classes and interfaces. To obtain a remote origin, the following code can be used:

### ↔ Sample Code

```

TRY.
    DATA(lo_rfc_destination) =
cl_rfc_destination_provider=>create_by_comm_arrangement( 'MY_COMM_ARRANGEMENT'
).
    CATCH cx_rfc_dest_provider_error INTO DATA(lx_rfc_dest_provider_error).
        " Handle exception.
ENDTRY.

DATA(lo_remote_origin) = xco_cp_abap_objects=>origin-
>remote( lo_rfc_destination ).
```

### ① Note

#### Authorization requirements for remote reading

Note that the user used for the system identified by the RFC destination needs to have the following authorizations in order to support the remote reading of classes and interfaces:

- S\_RFC with RFC\_TYPE = Function module, RFC\_NAME = SEO\_CLASS\_TYPEINFO\_GETRFC, SEO\_INTERFACE\_TYPEINFO\_GETRFC and ACTVT = Execute or alternatively for the complete function group, meaning
- S\_RFC with RFC\_TYPE = Function group, RFC\_NAME = SEO\_REMOTE and ACTVT = Execute

Once the origin has been obtained, it can be used for any of the exposed GET\_\* methods as part of the Read APIs for classes and interfaces. The code sample below illustrates how the public methods of the active version of a class can be read out for a remote destination:

### ↔ Sample Code

```

DATA(lt_public_methods) = xco_cp_abap=>class( 'MY_REMOTE_CLASS' )->definition-
>section-public->components->method->all->get(
    io_version = xco_cp_abap_objects=>version->active
```

```

    io_origin = lo_remote_origin
).

LOOP AT lt_public_methods INTO DATA(lo_public_method).
  " The name of the public method.
  DATA(lv_public_method_name) = lo_public_method->name.

  " The short description of the method.
  DATA(lv_short_description) = ls_public_method-short_description.

  " The indicator whether the method is ABSTRACT.
  DATA(lv_abstract_indicator) = ls_public_method-abstract_indicator.

  " The indicator whether the method is FINAL.
  DATA(lv_final_indicator) = ls_public_method-final_indicator.

  " The indicator whether the method is a REDEFINITION.
  DATA(lv_redefinition_indicator) = ls_public_method-redefinition_indicator.

  " A structure containing the AMDP attributes of the method (definition).
  DATA(ls_amdp) = ls_public_method-amdp.
ENDLOOP.

```

Similarly, using the same LO\_REMOTE\_ORIGIN as in the example above, we can read the constants of the inactive version of a remote interface:

#### ↔ Sample Code

```

DATA(lt_constants) = xco_cp_abap=>interface( 'MY_REMOTE_INTERFACE' )-
>components->constant->all->get(
  io_version = xco_cp_abap_objects=>version->inactive
  io_origin = lo_remote_origin
).

LOOP AT lt_constants INTO DATA(lo_constant).
  " The name of the constant.
  DATA(lv_constant_name) = lo_constant->name.

  DATA(ls_constant) = lo_constant->content( xco_cp_abap_objects=>version-
>inactive )->get( lo_remote_origin ).

  " The typing method of the constant.
  DATA(lo_typing_method) = ls_constant-typing_method.

  " The typing definition of the constant.
  DATA(lo_typing_definition) = ls_constant-typing_definition.

  " The value of the constant
  DATA(lv_value) = ls_constant-value.
ENDLOOP.

```

### 4.2.11.16.3.3 Generation APIs

The XCO Generation APIs are the part of the XCO library that allows the programmatic creation, update and deletion of ABAP repository objects. It consists of high-level and strongly typed APIs for the following objects types:

- BDEF (Behavior Definitions)

- CLAS (Classes)
- DCLS (Access Controls)
- DDLS (Data Definitions)
- DDLX (Metadata Extensions)
- DESD (Logical External Schema)
- DEVC (Packages)
- DOMA (Domains)
- DRTY (CDS Type Definitions)
- DTEL (Data Elements)
- EVTB (Event Bindings)
- FUGR (Function Groups)
- INTF (Interfaces)
- MSAG (Message Classes)
- NONT (SAP Object Node Types)
- RONT (SAP Object Types)
- SRVB (Service Bindings)
- SRVD (Service Definitions)
- TABL (Structures, Database Tables and Global Temporary Tables)
- TTYP (Table Types)
- XSLT (Transformations)

The following kinds of operations are supported:

- POST Create the object according to a provided specification
- PUT Create or update the object according to a provided specification
- PATCH Update the object according to a provided specification
- DELETE Delete the object

The matrix below shows which kinds of operations are supported for which object types:

	PUT	POST	PATCH	DELETE
BDEF	X			X
CLAS	X		X	X
DCLS	X			X
DDLS	X			X
DDLX	X			X
DESD	X			X
DEVC	X			X
DOMA	X		X	X
DRTY	X			X
DTEL	X		X	X
EVTB	X		X	X

	PUT	POST	PATCH	DELETE
FUGR		X	X	X
INTF	X		X	X
MSAG	X		X	X
NONT	X		X	X
RONT	X		X	X
SRVB	X			X
SRVD	X			X
TABL	X			X
TTYP	X			X
XSLT	X			X

### 4.2.11.16.3.3.1 Design of the XCO Generation APIs

As with the whole XCO library, the design of the XCO Generation APIs is governed by core principles:

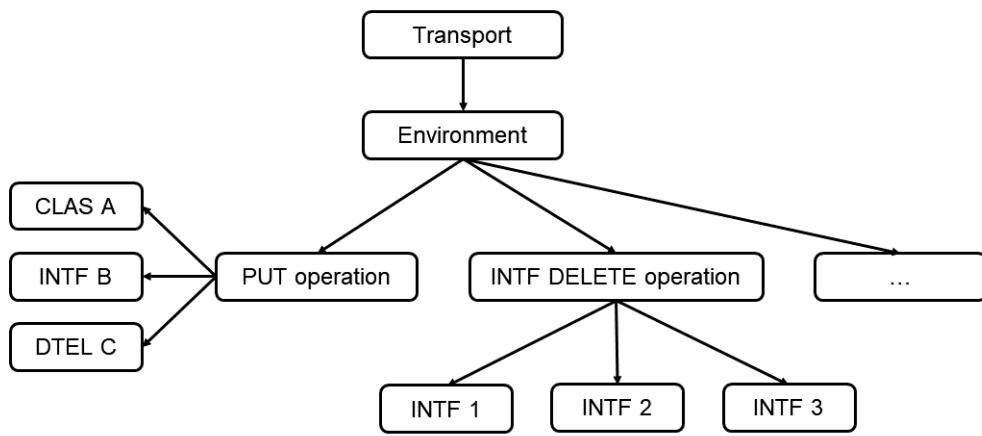
- **Environment**

The prerequisite for building and executing operations is a generation environment. A generation environment is backed by a transport which must be in status 'Modifiable' and is obtained via the XCO\_CP\_GENERATION API.

 Sample Code

```
" In this example, X08K123456 would need to be a modifiable Workbench
" transport.
DATA(lo_environment) = xco_cp_generation->environment-
>dev_system( 'X08K123456' ).
DATA(lo_put_operation) = lo_environment->create_put_operation( ).
lo_put_operation->for-dtel->add_object( '...' ).
"
" ...
```

The environment acts as the entry point to the type-specific Generation APIs which allow to build and successively execute PUT and DELETE operations. All system changes caused by an operation created for an environment will be written to the environment's transport.



- Operations

Operations consist of a set of objects (identified by their type and name) and allow to perform an action for these objects (determined by the kind of operation).

#### *PUT operations*

Upon execution, a PUT operation either creates or updates its objects according to provided specifications (depending on whether the object already exists in the system). Each object type defines a form-based specification tailored to the specific attributes of the object type which is used to describe the content of each object of the PUT operation. As a PUT operation potentially creates new objects a valid package must be provided for all objects without a package. This package is used when the object is newly created but is ignored when the object already exists and is only updated.

#### ↔ Sample Code

```

DATA(lo_put_operation) = lo_environment->create_put_operation( ).
DATA(lo_specification) = lo_put_operation->for-dtel-
>add_object( 'ZDATA_ELEMENT'
    )->set_package( 'ZPACKAGE'
    )->create_form_specification( ).
lo_specification->set_short_description( 'My generated data element' ).
lo_specification->set_data_type( xco_cp_abap_dictionary->built_in_type-
>char( 10 ) ).
lo_specification->field_label-short->set_text( 'ID' ).
lo_specification->field_label-medium->set_text( 'Identifier' ).
lo_specification->field_label-long->set_text( 'User identifier' ).
lo_specification->field_label-heading->set_text( 'User identifier' ).
lo_put_operation->execute( ).
```

Object types that support inactive versions can be combined into a single PUT operation. Once the objects have either been created or updated (in inactive version) all the objects of the PUT operation will be activated together in a single mass activation. This provides support for interdependencies (even circular ones) between the objects of a PUT operation.

It is furthermore possible to influence the execution of a PUT operation with so-called options. Currently the following options are offered:

- Skip activation: Create or update the objects of the PUT operation but do not perform a mass activation afterwards

Using an option is as simple as supplying it to the EXECUTE method invocation of the corresponding PUT operation:

#### ↔ Sample Code

```
lo_put_operation->execute( VALUE  
#( ( xco_cp_generation=>put_operation_option->skip_activation ) ) ).
```

#### *POST operations*

The objects on a POST operation consist of a name choice (IF\_XCO\_NAME\_CHOICE) and a specification. In contrast to PUT operations, objects are not explicitly identified by their name, but instead a name choice is provided which will be evaluated upon execution of the POST operation. By providing a name choice instead of a fixed name it is possible to potentially pick a fallback name in case the preferred name is already taken. However, this is not mandatory and it is also possible to provide a fixed name, e.g. for the creation of function group:

```
DATA(lo_post_operation) = lo_environment->for-fugr->create_post_operation( ).  
DATA(lo_fixed_name_choice) = xco_cp_name=>choice->fixed( 'ZMY_FNCTN_GRP' ).  
lo_post_operation->add_object( lo_fixed_name_choice  
    )->set_package( 'ZLOCAL'  
    )->create_form_specification( ).  
"  
    ...  
lo_post_operation->execute( ).
```

In any case, upon execution the name choices for all the objects on the POST operation are resolved. The operation will only proceed if all name choices can be resolved, i.e. a name can be determined for which no object already exists in the ABAP repository. If all name choices were resolved successfully the semantics of a POST operation are those of a PUT operation where the objects do not exist prior to the execution.

#### *PATCH operations*

Upon execution, a PATCH operation will update its objects according to the changes provided in the specifications. The available changes depend on the object type.

For message classes (MSAG) it is e.g. possible to insert, update, modify or delete messages of the message class. The semantics are as follows:

- **INSERT** Add a new part to the object if it does not exist yet. If it already exists, no action will be performed.
- **UPDATE** Update an existing part of the object. If the part does not exist, no action will be performed.
- **MODIFY** Modify the specified part of the object. If the part does not exist, it will be created according to the specification, otherwise it will be updated.
- **DELETE** Delete the specified part of the object. If the specified part does not exist no action will be performed

#### *DELETE operations*

Upon execution, a DELETE operation will delete all its objects (in case they exist in the system). DELETE operations are always built and executed for a given type and it is the responsibility of the caller to ensure that objects of different types are deleted in the correct order (e.g. a domain still in use by a data element cannot be deleted; the data element must be deleted first).

#### *COMMIT behavior of operations*

The execution of an operation will always execute a COMMIT WORK.

#### *Errors during operation execution*

If the execution of an operation fails, the following exceptions are used to communicate the failure:

- CX\_XCO\_GEN\_PUT\_EXCEPTION for a failed PUT operation
- CX\_XCO\_GEN\_DELETE\_EXCEPTION for a failed DELETE operation

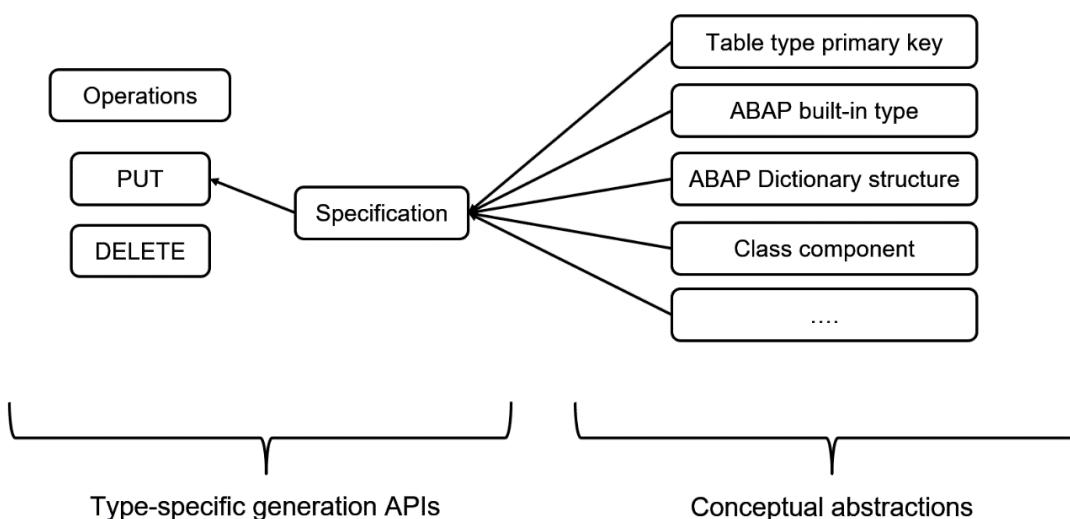
- CX\_XCO\_GEN\_PATCH\_EXCEPTION for a failed PATCH operation

Each exception provides access to the findings, i.e. all the messages and which object they belong to, that were encountered when attempting to execute the operation.

In case the mass activation of a PUT operation fails, the already existing inactive versions will not be deleted.

- **Conceptual Abstractions**

The overall design and architecture of the XCO Generation functionality is based on the differentiation between conceptual abstractions and type-specific generation APIs.



Conceptual abstractions represent entities in the ABAP system which can be used in conjunction with different object types. For example, a built-in type of the ABAP Dictionary may be used as the format for a domain (DOMA), the data type of a data element (DTEL) or the row type of a table type (TTYP). Conceptual abstractions create a common language which can be used across different object types in the context of generation.

#### ↔ Sample Code

```

DATA(lo_char_10) = xco_cp_abap_dictionary->built_in_type->char( 10 ).
lo_doma_specification->set_format( lo_char_10 ).
lo_dtel_specification->set_data_type( lo_char_10 ).
lo_ttyp_specification->set_row_type( lo_char_10 ).
```

- **4. Templates**

Instead of manually filling a form specification, templates can be used to prefill an empty form specification from an existing ABAP Repository object. Currently the following object types support templates:

- DOMA (Domain)
- DTEL (Data element)
- TABL (Database tables and structures)
- TTYP (Table types)
- DDLS (Data Definitions)

Using templates, it is e.g. possible to easily copy an existing object. The following code sample uses the active version of domain ZSOURCE\_DOMAIN as a template and will create (resp. update) domain ZCOPIED\_DOMAIN to reflect the content of ZSOURCE\_DOMAIN:

#### ↔ Sample Code

```
DATA(lo_template) = xco_cp_generation_doma->template->for_domain(
    iv_name      = 'ZSOURCE_DOMAIN'
    io_read_state = xco_cp_abap_dictionary->object_read_state->active_version
).
DATA(lo_put_operation) = lo_environment->create_put_operation( ).
lo_put_operation->for-doma->add_object( 'ZCOPIED_DOMAIN'
    )->set_package( 'ZPACKAGE'
    )->set_template( lo_template ).
lo_put_operation->execute( ).
```

### 4.2.11.16.3.3.2 RAP BO Generation

The following code sample illustrates how a complete RAP business object, starting from domains for fixed values all the way up to a service binding, can be generated with the XCO Generation APIs:

#### ↔ Sample Code

```
! Code sample for executing PUT operations to generate a full RAP business
object.
!
! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
! with an existing package and a modifiable Workbench transport matching the
transport
! target of the package.
CLASS zcl_xco_doc_cp_cs_gen_put DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.
PROTECTED SECTION.
METHODS:
    main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:
    co_package          TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
    co_transport        TYPE sxco_transport VALUE 'X08K900164',
    co_doma_status     TYPE sxco_ad_object_name VALUE
'ZXCO_CP_VR_STATUS',           co_dtel_status      TYPE sxco_ad_object_name VALUE
'ZXCO_CP_VR_STATUS',           co_tabl_dbt_name   TYPE sxco_dbt_object_name VALUE
'ZXCO_CP_VAC_REQ',            co_ddls_interface_name  TYPE sxco_cds_object_name VALUE
'ZXCO_CP_I_VACATION_REQUEST', co_ddls_consumption_name  TYPE sxco_cds_object_name VALUE
'ZXCO_CP_C_VACATION_REQUEST', co_ddlx_name       TYPE sxco_cds_object_name VALUE
'ZXCO_CP_VACATION_REQUEST_EXT',co_dcls_name       TYPE sxco_cds_object_name VALUE
'ZXCO_CP_VACTION_REQUEST_AC', co_clas_name       TYPE sxco_ad_object_name VALUE
'ZXCO_CP_BP_VACATION_REQUEST',
```

```

        co_srvd_name           TYPE sxco_srvd_object_name VALUE
'ZXCO_CP_VAC_REQ_SRVD',
        co_srvb_name           TYPE sxco_srvb_object_name VALUE
'ZXCO_CP_VAC_REQ_SRVB'.
DATA:
mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.
METHODS:
add_doma
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_dtel
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_tabl_dbt
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_ddls
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_ddlx
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_dcls
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_bdef
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_clas
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_srvd
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
put_srvb.
ENDCLASS.
CLASS zcl_xco_doc_cp_cs_gen_put IMPLEMENTATION.
METHOD constructor.
super->constructor( ).
mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.
METHOD main.
DATA(lo_put_operation) = mo_environment->create_put_operation( ).
add_doma( lo_put_operation ).
add_dtel( lo_put_operation ).
add_tabl_dbt( lo_put_operation ).
add_ddls( lo_put_operation ).
add_ddlx( lo_put_operation ).
add_dcls( lo_put_operation ).
add_bdef( lo_put_operation ).
add_clas( lo_put_operation ).
add_srvd( lo_put_operation ).
lo_put_operation->execute( ).
put_srvb( ).
out->write( |PUT operations executed successfully.| ) .
ENDMETHOD.
METHOD add_doma.
DATA(lo_specification) = io_put_operation->for-domains-
>add_object( co_doma_status
  )->set_package( co_package
  )->create_form_specification( ).
lo_specification->set_short_description( 'Vacation request status' ).
lo_specification->set_format( xco_cp_abap_dictionary=>built_in_type-
>char( 10 ) ).
lo_specification->fixed_values->add_fixed_value( 'APPROVED'
  )->set_description( 'Approved' ).
lo_specification->fixed_values->add_fixed_value( 'DECLINED'

```

```

        )->set_description( 'Declined' ).
ENDMETHOD.

METHOD add_dtel.
    DATA(lo_specification) = io_put_operation->for-dtel-
>add_object( co_dtel_status
        )->set_package( co_package
        )->create_form_specification( ).
    lo_specification->set_short_description( 'Vacation request status' ).
    lo_specification-
>set_data_type( xco_cp_abap_dictionary=>domain( co_doma_status ) ).
ENDMETHOD.

METHOD add_tabl_dbt.
    DATA(lo_specification) = io_put_operation->for-tabl-for-database_table-
>add_object( co_tabl_dbt_name
        )->set_package( co_package
        )->create_form_specification( ).
    lo_specification->set_short_description( 'Vacation request'
        )->set_delivery_class( xco_cp_database_table=>delivery_class->a
        )->set_data_maintenance( xco_cp_database_table=>data_maintenance-
>allowed ).
    lo_specification->add_field( 'CLIENT' )-
>set_type( xco_cp_abap_dictionary=>built_in_type->clnt
        )->set_key_indicator(
        )->set_not_null( ).
    lo_specification->add_field( 'IDENTIFIER' )-
>set_type( xco_cp_abap_dictionary=>built_in_type->char( 30 )
        )->set_key_indicator(
        )->set_not_null( ).
    lo_specification->add_field( 'START_DATE' )-
>set_type( xco_cp_abap_dictionary=>built_in_type->date )..
    lo_specification->add_field( 'END_DATE' )-
>set_type( xco_cp_abap_dictionary=>built_in_type->date )..
    lo_specification->add_field( 'STATUS' )-
>set_type( xco_cp_abap_dictionary=>data_element( co_dtel_status ) ).
    lo_specification->add_field( 'IS_ACTIVE' )-
>set_type( xco_cp_abap_dictionary=>data_element( 'ABAP_BOOLEAN' ) ).
ENDMETHOD.

METHOD add_ddls.
    DATA(lo_interface_specification) = io_put_operation->for-ddls-
>add_object( co_ddls_interface_name
        )->set_package( co_package
        )->create_form_specification( ).
    DATA(lo_view_entity) = lo_interface_specification-
>set_short_description( 'Vacation request'
        )->add_view_entity( .
    lo_view_entity->set_root( )->data_source->set_view_entity( CONV
#( co_tabl_dbt_name ) ).
    lo_view_entity->set_where( xco_cp_ddl=>field( 'is_active' )-
>eq( xco_cp_ddl=>literal->character( 'X' ) ) ).
    " View annotations.
    lo_view_entity->add_annotation( 'AccessControl.authorizationCheck' )-
>value->build( )->add_enum( 'CHECK' ).
    lo_view_entity->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( 'Vacation request view entity' ).
    lo_view_entity->add_annotation( 'Metadata.allowExtensions' )->value-
>build( )->add_boolean( abap_true ).
    " Add fields.
    DATA(lo_identifier)
= lo_view_entity->add_field( xco_cp_ddl=>field( 'identifier' ) )->set_key( )-
>set_alias( 'Identifier' ).
    lo_identifier->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( 'Identifier' ).
    lo_view_entity->add_field( xco_cp_ddl=>field( 'start_date' ) )-
>set_alias( 'StartDate' ).
    lo_view_entity->add_field( xco_cp_ddl=>field( 'end_date' ) )-
>set_alias( 'EndDate' ).
    lo_view_entity->add_field( xco_cp_ddl=>field( 'status' ) )-
>set_alias( 'Status' ).
```

```

        DATA(lo_consumption_specification) = io_put_operation->for-ddls-
>add_object( co_ddls_consumption_name
    )->set_package( co_package
    )->create_form_specification( ).
        DATA(lo_projection_view) = lo_consumption_specification-
>set_short_description( 'Vacation request projection'
    )->add_projection_view( ).
        lo_projection_view->set_root( ).
        lo_projection_view->data_source-
>set_view_entity( co_ddls_interface_name ).
        lo_projection_view->add_field( xco_cp_ddl=>field( 'Identifier' ) )-
>set_key( ).
        lo_projection_view->add_field( xco_cp_ddl=>field( 'StartDate' ) ).
        lo_projection_view->add_field( xco_cp_ddl=>field( 'EndDate' ) ).
        lo_projection_view->add_field( xco_cp_ddl=>field( 'Status' ) ).
ENDMETHOD.

METHOD add_ddlx.
    DATA(lo_specification) = io_put_operation->for-ddlx-
>add_object( co_ddlx_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_specification->set_short_description( 'Vacation request extension'
    )->set_layer( xco_cp_metadata_extension=>layer->customer
    )->set_view( co_ddls_interface_name ).

    " UI annotations.

    TYPES:
        BEGIN OF ts_field,
            name TYPE sxco_cds_field_name,
            label TYPE string,
        END OF ts_field,
        tt_field TYPE STANDARD TABLE OF ts_field WITH EMPTY KEY.
    DATA(lt_fields) = VALUE tt_field(
        ( name = 'Identifier' label = 'Identifier' )
        ( name = 'StartDate' label = 'Start Date' )
        ( name = 'EndDate' label = 'End Date' )
        ( name = 'Status' label = 'Status' )
    ).
    LOOP AT lt_fields INTO DATA(ls_field).
        DATA(lv_position) = sy-tabix * 10.
        DATA(lo_field) = lo_specification->add_field( ls_field-name ).
        lo_field->add_annotation( 'UI.lineItem' )->value->build(
            )->begin_array(
                )->begin_record(
                    )->add_member( 'position' )->add_number( lv_position
                    )->add_member( 'label' )->add_string( ls_field-label
                )->end_record(
            )->end_array( ).

        lo_field->add_annotation( 'UI.identification' )->value->build(
            )->begin_array(
                )->begin_record(
                    )->add_member( 'position' )->add_number( lv_position
                    )->add_member( 'label' )->add_string( ls_field-label
                )->end_record(
            )->end_array( ).

    ENDLOOP.
ENDMETHOD.

METHOD add_dcls.
    DATA(lo_specification) = io_put_operation->for-dcls-
>add_object( co_dcls_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_specification->set_short_description( 'Vacation request access
control' ).

    DATA(lo_role) = lo_specification->add_role( ).
    lo_role->add_annotation( 'MappingRole' )->value->build(
        )->add_boolean( abap_true ).

    lo_role->add_access_rule( co_ddls_interface_name
        )->set_where( xco_cp_dcl=>field( 'Identifier' )->is_not_initial( ) ).
```

```

ENDMETHOD.
METHOD add_bdef.
    DATA(lo_interface_specification) = io_put_operation->for-bdef-
>add_object( co_ddls_interface_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_interface_specification->set_short_description( 'Vacation request
behavior'
    )-
>set_implementation_type( xco_cp_behavior_definition=>implementation_type-
>managed
    )->set_implementation_class( co_clas_name ).
    DATA(lo_interface_behavior) = lo_interface_specification->add_behavior( ).
    " Characteristics.
    lo_interface_behavior->characteristics-
>set_persistent_table( co_tabl_dbt_name
    )->lock->set_master( ).
    " Fields.
    lo_interface_behavior->add_mapping_for( co_tabl_dbt_name )-
>set_field_mapping( VALUE #(
        ( cds_view_field = 'Identifier' dbtable_field = 'identifier' )
        ( cds_view_field = 'StartDate' dbtable_field = 'start_date' )
        ( cds_view_field = 'EndDate' dbtable_field = 'end_date' )
        ( cds_view_field = 'Status' dbtable_field = 'status' )
    )).
    " Standard operations.
    lo_interface_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>create ).
    lo_interface_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>update ).
    lo_interface_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>delete ).
    " Action.
    DATA(lo_action) = lo_interface_behavior->add_action( 'approve' ).
    lo_action->set_external_name( 'Approve' ).
    lo_action->result->set_cardinality( xco_cp_cds=>cardinality->one )-
>set_self( ).
    DATA(lo_consumption_specification) = io_put_operation->for-bdef-
>add_object( co_ddls_consumption_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_consumption_specification->set_short_description( 'Vacation request
behavior projection'
    )-
>set_implementation_type( xco_cp_behavior_definition=>implementation_type-
>projection ).
    DATA(lo_consumption_behavior) = lo_consumption_specification-
>add_behavior( ).
    lo_consumption_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>create
        )->set_use( ).
    lo_consumption_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>update
        )->set_use( ).
    lo_consumption_behavior-
>add_standard_operation( xco_cp_behavior_definition=>standard_operation-
>delete
        )->set_use( ).
ENDMETHOD.
METHOD add_clas.
    DATA(lo_specification) = io_put_operation->for-clas-
>add_object( co_clas_name
    )->set_package( co_package

```

```

        )->create_form_specification( ).
    lo_specification->set_short_description( 'Behavior implementation' ).
    lo_specification->definition->set_abstract(
        )->set_for_behavior_of( co_ddls_interface_name ).
    DATA(lo_handler) = lo_specification->add_local_class( 'LCL_HANDLER' ).
    lo_handler->definition->set_superclass( 'CL_ABAP_BEHAVIOR_HANDLER' ).
    " Action implementation.
    DATA(lo_publish) = lo_handler->definition->section-private-
>add_method( 'APPROVE' ).
    lo_publish->behavior_implementation->set_for_modify(
        )->set_result( co_ddls_interface_name ).
    lo_publish->add_importing_parameter( 'IT_VACATION_REQUESTS'
        )->behavior_implementation->set_for_action(
            iv_entity_name = co_ddls_interface_name
            iv_action_name = 'approve'
        ).
    lo_handler->implementation->add_method( 'APPROVE' ).
ENDMETHOD.

METHOD add_srwd.
    DATA(lo_specification) = io_put_operation->for-srwd-
>add_object( co_srwd_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_specification->set_short_description( 'Service definition' ).
    lo_specification->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( 'Vacation request service definition' ).
    lo_specification->add_exposure( co_ddls_consumption_name )-
>set_alias( 'VacationRequest' ).
ENDMETHOD.

METHOD put_srbd.
    DATA(lo_put_operation) = mo_environment->for-srbd-
>create_put_operation( ).
    DATA(lo_specification) = lo_put_operation->add_object( co_srbd_name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_specification->set_short_description( 'Service binding' ).
    lo_specification->set_binding_type( xco_cp_service_binding=>binding_type-
>odata_v2_ui ).
    lo_specification->add_service( )->add_version( '0001' )-
>set_service_definition( co_srbd_name ).
    lo_put_operation->execute( ).
ENDMETHOD.

ENDCLASS.

```

All objects generated with the previous code sample can also be deleted again:

#### ↳ Sample Code

```

#! Code sample for executing DELETE operations to delete all objects of a
full RAP business
#! object.
"!
#! IMPORTANT: The value of the constant CO_TRANSPORT must be replaced with a
modifiable Workbench
#! transport matching the transport target of the package containing the
objects.
CLASS zcl_xco_doc_cp_cs_gen_delete DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.
PROTECTED SECTION.
METHODS:
    main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:

```

```

        co_transport          TYPE sxco_transport VALUE 'X08K900164',
        co_doma_status        TYPE sxco_ad_object_name VALUE
'ZXCO_CP_VR_STATUS',
        co_dtel_status        TYPE sxco_ad_object_name VALUE
'ZXCO_CP_VR_STATUS',
        co_tabl_dbt_name      TYPE sxco_dbt_object_name VALUE
'ZXCO_CP_VAC_REQ',
        co_ddls_interface_name TYPE sxco_cds_object_name VALUE
'ZXCO_CP_I_VACATION_REQUEST',
        co_ddls_consumption_name TYPE sxco_cds_object_name VALUE
'ZXCO_CP_C_VACATION_REQUEST',
        co_ddlx_name          TYPE sxco_cds_object_name VALUE
'ZXCO_CP_VACATION_REQUEST_EXT',
        co_dcls_name          TYPE sxco_cds_object_name VALUE
'ZXCO_CP_VACTION_REQUEST_AC',
        co_clas_name          TYPE sxco_ad_object_name VALUE
'ZXCO_CP_BP_VACATION_REQUEST',
        co_srwd_name          TYPE sxco_srwd_object_name VALUE
'ZXCO_CP_VAC_REQ_SRVD',
        co_srvb_name          TYPE sxco_srvb_object_name VALUE
'ZXCO_CP_VAC_REQ_SRVB'.
DATA:
mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.

METHODS:
    delete_srvb,
    delete_srwd,
    delete_clas,
    delete_bdef,
    delete_dcls,
    delete_ddlx,
    delete_ddls,
    delete_tabl_dbt,
    delete_dtel,
    delete_doma.

ENDCLASS.
CLASS zcl_xco_doc_cp_cs_gen_delete IMPLEMENTATION.
METHOD constructor.
    super->constructor( ).
    mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.
METHOD main.
    DATA(lo_put_operation) = mo_environment->create_put_operation( ).
    delete_srvb( ).
    delete_srwd( ).
    delete_clas( ).
    delete_bdef( ).
    delete_dcls( ).
    delete_ddlx( ).
    delete_ddls( ).
    delete_tabl_dbt( ).
    delete_dtell( ).
    delete_doma( ).
    out->write( |DELETE operations executed successfully.| ).
ENDMETHOD.
METHOD delete_srvb.
    DATA(lo_delete_operation) = mo_environment->for-srvb-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_srvb_name ).
    lo_delete_operation->execute( ).
ENDMETHOD.
METHOD delete_srwd.
    DATA(lo_delete_operation) = mo_environment->for-srwd-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_srwd_name ).
    lo_delete_operation->execute( ).
ENDMETHOD.
METHOD delete_clas.

```

```

        DATA(lo_delete_operation) = mo_environment->for-clas-
>create_delete_operation( ).
        lo_delete_operation->add_object( co_clas_name ).
        lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_bdef.
    DATA(lo_delete_operation) = mo_environment->for-bdef-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_ddls_consumption_name ).
    lo_delete_operation->add_object( co_ddls_interface_name ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_dcls.
    DATA(lo_delete_operation) = mo_environment->for-dcls-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_dcls_name ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_ddlx.
    DATA(lo_delete_operation) = mo_environment->for-ddlx-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_ddlx_name ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_ddls.
    DATA(lo_delete_operation) = mo_environment->for-ddls-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_ddls_consumption_name ).
    lo_delete_operation->add_object( co_ddls_interface_name ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_tabl_dbt.
    DATA(lo_delete_operation) = mo_environment->for-tabl-for-database_table-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_tabl_dbt_name ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_dtel.
    DATA(lo_delete_operation) = mo_environment->for-dtel-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_dtel_status ).
    lo_delete_operation->execute( ).

ENDMETHOD.

METHOD delete_doma.
    DATA(lo_delete_operation) = mo_environment->for-doma-
>create_delete_operation( ).
    lo_delete_operation->add_object( co_doma_status ).
    lo_delete_operation->execute( ).

ENDMETHOD.

ENDCLASS.

```

## 4.2.11.16.3.3.3 Generation Error Handling

Each type of operation within the XCO Generation APIs defines its own exception class that can be used to deal with errors that were encountered during the execution of the operation:

- CX\_XCO\_GEN\_POST\_EXCEPTION for POST operations
- CX\_XCO\_GEN\_PUT\_EXCEPTION for PUT operations
- CX\_XCO\_GEN\_PATCH\_EXCEPTION for PATCH operations

- CX\_XCO\_GEN\_DELETE\_EXCEPTION for DELETE operations

The following code sample shows how the findings of a failed PUT operation can be extracted:

### ↳ Sample Code

```

"! Code sample for handling errors of a PUT operation.
"!
"! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
"! with an existing package and a modifiable Workbench transport matching the
transport
"! target of the package.
CLASS zcl_xco_doc_cp_cs_gen_error DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
  constructor.
PROTECTED SECTION.
METHODS:
  main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:
  co_package  TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
  co_transport TYPE sxco_transport VALUE 'X08K900164',
  co_doma_name TYPE sxco_ad_object_name VALUE
'ZXCO_CP_NON_EXISTENT_DOMAIN',
  co_dtel_name TYPE sxco_ad_object_name VALUE 'ZXCO_CP_INVALID_DTEL',
  co_ttyp_name TYPE sxco_ad_object_name VALUE 'ZXCO_CP_INVALID_TTYP'.
DATA:
  mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.
ENDCLASS.
CLASS zcl_xco_doc_cp_cs_gen_error IMPLEMENTATION.
METHOD constructor.
  super->constructor( ).
  mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.
METHOD main.
  DATA(lo_put_operation) = mo_environment->create_put_operation( ).
  " Add an invalid data element
  DATA(lo_dtel_specification) = lo_put_operation->for-dtel-
>add_object( co_dtel_name
    )->set_package( co_package
    )->create_form_specification( ).
  lo_dtel_specification->set_short_description( 'My data element' ).
  lo_dtel_specification-
>set_data_type( xco_cp_abap_dictionary=>domain( co_doma_name ) ).
  " Add an invalid table type
  DATA(lo_ttyp_specification) = lo_put_operation->for-ttyp-
>add_object( co_ttyp_name
    )->set_package( co_package
    )->create_form_specification( ).
  lo_ttyp_specification->set_short_description( 'My table type' ).
  lo_ttyp_specification-
>set_row_type( xco_cp_abap_dictionary=>data_element( co_dtel_name ) ).
TRY.
  lo_put_operation->execute( ).
  CATCH cx_xco_gen_put_exception INTO DATA(lx_put_exception).
  " Get all findings.
  out->plain->write( |Findings:| ).
  out->write_news( lx_put_exception ).
  " Get only the TTYP findings.
  out->plain->write( |Table type findings:| ).
  out->write_news( lx_put_exception->findings->for->ttyp ).
ENDTRY.
ENDMETHOD.

```

```
ENDCLASS.
```

#### 4.2.11.16.3.3.4 Generating and Reading CDS Abstract Entities

The following code sample illustrates how abstract entities can be generated and read with the XCO Library:

```
"! <p class="shorttext synchronized" lang="EN">
"!   XCO Documentation CP: Abstract entities
"! </p>
"!
"! Code sample for generating and reading CDS abstract entities.
"!
"! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
"! with an existing package and a modifiable Workbench transport matching the
transport
"! target of the package.
CLASS zcl_xco_doc_cp_cs_abs_ent DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PUBLIC SECTION.
    METHODS:
      constructor.
  PROTECTED SECTION.
    METHODS:
      main REDEFINITION.
  PRIVATE SECTION.
    CONSTANTS:
      co_package          TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
      co_transport        TYPE sxco_transport VALUE 'X08K900164',
      co_abstract_entity_name TYPE sxco_ao_object_name VALUE 'ZXCO_DEMO_PRODUCT'.
    DATA:
      mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.
    METHODS:
      put_ddls.
  ENDCLASS.
CLASS zcl_xco_doc_cp_cs_abs_ent IMPLEMENTATION.
  METHOD constructor.
    super->constructor( ).
    mo_environment = xco_cp_generation=>environment->dev_system( co_transport ).
  ENDMETHOD.
  METHOD main.
    put_ddls( ).
    out->write( |PUT operation executed successfully.| ).
    DATA(lo_abstract_entity) =
      xco_cp_cds=>abstract_entity( co_abstract_entity_name ).
    DATA(ls_abstract_entity) = lo_abstract_entity->content( )->get( ).
    out->write( ls_abstract_entity ).
    LOOP AT lo_abstract_entity->fields->all->get( ) INTO DATA(lo_field).
      out->write( |Field { lo_field->name }:| ).
      DATA(ls_field) = lo_field->content( )->get( ).
      out->write( ls_field ).
    ENDLOOP.
  ENDMETHOD.
  METHOD put_ddls.
    DATA(lo_put_operation) = mo_environment->create_put_operation( ).
    DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( co_abstract_entity_name
    )->set_package( co_package
    )->create_form_specification( ).
    DATA(lo_abstract_entity) = lo_form_specification-
>set_short_description( 'Product'
```

```

) ->add_abstract_entity( ).

DATA(lo_parameter) = lo_abstract_entity->add_parameter( 'p_language'
) ->set_data_type( xco_cp_abap_dictionary=>built_in_type->lang ).

DATA(lo_id_field) = lo_abstract_entity-
>add_field( xco_cp_ddl=>field( 'ID' ) ).

lo_id_field->set_key( ).

lo_id_field->set_type( xco_cp_abap_dictionary=>built_in_type->char( 30 ) ).

lo_id_field->add_annotation( 'UI.lineItem' )->value->build(
) ->begin_array(
) ->begin_record(
) ->add_member( 'position' )->add_number( 1
) ->add_member( 'label' )->add_string( 'Product'
) ->end_record(
) ->end_array( ).

DATA(lo_description_field) = lo_abstract_entity-
>add_field( xco_cp_ddl=>field( 'description' ) ).

lo_description_field->set_type( xco_cp_abap_dictionary=>built_in_type-
>char( 60 ) ).

lo_description_field->add_annotation( 'UI.lineItem' )->value->build(
) ->begin_array(
) ->begin_record(
) ->add_member( 'position' )->add_number( 2
) ->add_member( 'label' )->add_string( 'Description'
) ->end_record(
) ->end_array( ).

lo_put_operation->execute( ).

ENDMETHOD.

ENDCLASS.

```

## 4.2.11.16.3.3.5 Generating and Reading Global Temporary Tables

The following code sample illustrates how global temporary tables can be generated and read with the XCO Library:

```

"! <p class="shorttext synchronized" lang="EN">
"!   XCO Documentation CP: Global temporary tables
"! </p>
"!
"! Code sample for generating and reading global temporary tables.
"!
"! IMPORTANT: The value of the constant CO_PACKAGE must be replaced with an
existing
"! development package.
CLASS zcl_xco_doc_cp_cs_abs_ent DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PUBLIC SECTION.
  METHODS:
    constructor.

  PROTECTED SECTION.
  METHODS:
    main REDEFINITION.

  PRIVATE SECTION.
  CONSTANTS:
    co_package           TYPE sxco_package VALUE 'ZMY_PACKAGE',
    co_global_temporary_table_name TYPE sxco_gtt_object_name VALUE 'ZXCO_GTT'.

  DATA:

```

```

mo_global_temporary_table TYPE REF TO if_xco_global_temporary_table.

METHODS:
  put_global_temporary_table
    IMPORTING
      out TYPE REF TO if_xco_adt_classrun_out,
    get_generation_environment
      IMPORTING
        out
      RETURNING
        TYPE REF TO if_xco_adt_classrun_out
        VALUE(ro_generation_environment) TYPE REF TO
if_xco_cp_gen_env_dev_system,
  get_transport_request
    IMPORTING
      out
    RETURNING
      TYPE REF TO if_xco_adt_classrun_out
      VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_abs_ent IMPLEMENTATION.
  METHOD constructor.
    super->constructor( ).
    mo_global_temporary_table =
xco_cp_abap_dictionary->global_temporary_table( co_global_temporary_table_name ).
  ENDMETHOD.

  METHOD main.
    put_global_temporary_table( out ).
    out->write( |PUT operation executed successfully.| ).
    DATA(ls_global_temporary_table) = mo_global_temporary_table->content( )-
>get( ).
    out->write( ls_global_temporary_table ).

    LOOP AT mo_global_temporary_table->fields->all->get( ) INTO DATA(lo_field).
      out->write( |Field { lo_field->name }:| ).

      DATA(ls_field) = lo_field->content( )->get( ).
      out->write( ls_field ).
    ENDLOOP.
  ENDMETHOD.

  METHOD put_global_temporary_table.
    DATA(lo_put_operation) = get_generation_environment( out )-
>create_put_operation( ).

    DATA(lo_form_specification) = lo_put_operation->for-tabl-for-
global_temporary_table->add_object( mo_global_temporary_table->name
)-
>set_package( co_package
)-
>create_form_specification( ).
    lo_form_specification->set_short_description( 'Generated global temporary
table' ).

    lo_form_specification->add_field( 'KEY_FIELD'
)-
>set_type( xco_cp_abap_dictionary->built_in_type->char( 10 )
)-
>set_key_indicator(
)-
>set_not_null( ).

    lo_form_specification->add_field( 'DATA_FIELD'
)-
>set_type( xco_cp_abap_dictionary->built_in_type->char( 60 ) ).

    lo_put_operation->execute( ).
  ENDMETHOD.

  METHOD get_generation_environment.

```

```

DATA(lo_transport_request) = get_transport_request( out ).

    ro_generation_environment = xco_cp_generation=>environment-
>dev_system( lo_transport_request->value ).
ENDMETHOD.

METHOD get_transport_request.
    DATA(lo_lock) = mo_global_temporary_table->if_xco_cts_changeable~get_object(
        )->get_lock( ).

    IF lo_lock->exists( ) EQ abap_true.
        DATA(lv_transport) = lo_lock->get_transport( ).

        ro_transport_request = xco_cp_cts=>transport->for( lv_transport
            )->get_request( ).

        out->write( |Using transport request { ro_transport_request->value } |
(Global temporary table is already locked).| ).

    ELSE.
        DATA(lo_transport_target) = xco_cp_abap_repository=>package-
>for( co_package
        )->read(
        )-property-transport_layer->get_transport_target( ).

        ro_transport_request = xco_cp_cts=>transports-
>workbench( lo_transport_target->value
        )->create_request( 'Generated transport request' ).

        out->write( |Using newly generated transport request
{ ro_transport_request->value }.| ).
    ENDIF.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.16.3.3.6 Logic Generation

The following code sample illustrates the generation of classes and interfaces:

### ↔ Sample Code

```

"! <p class="shorttext synchronized" lang="EN">
"!   XCO Documentation CP: Logic generation
"! </p>
"!
"! Code sample for generating an interface along with a class that implements
it.
"!
"! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
"! with an existing package and a modifiable Workbench transport matching the
transport
"! target of the package.
CLASS zcl_xco_doc_cp_cs_gen_logic DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PUBLIC SECTION.
    METHODS:
      constructor.
  PROTECTED SECTION.
    METHODS:
      main REDEFINITION.
  PRIVATE SECTION.
    CONSTANTS:

```

```

co_package      TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
co_transport    TYPE sxco_transport VALUE 'X08K900164',
co_interface_name TYPE sxco_ao_object_name VALUE 'ZIF_CUSTOMER',
co_class_name   TYPE sxco_ad_object_name VALUE 'ZCL_CUSTOMER'.
DATA:
mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.
METHODS:
add_intf
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
add_clas
  IMPORTING
    io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put.
ENDCLASS.
CLASS zcl_xco_doc_cp_cs_gen_logic IMPLEMENTATION.
METHOD constructor.
super->constructor( ).
mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.
METHOD main.
DATA(lo_put_operation) = mo_environment->create_put_operation( ).
add_intf( lo_put_operation ).
add_clas( lo_put_operation ).
lo_put_operation->execute( ).
out->write( |PUT operation executed successfully.| ).
ENDMETHOD.
METHOD add_intf.
DATA(lo_specification) = io_put_operation->for-intf-
>add_object( co_interface_name
  )->set_package( co_package
  )->create_form_specification( ).
lo_specification->set_short_description( 'Customer' ).
lo_specification->add_type( 'TV_PRODUCT' )->for( xco_cp_abap=>type-
built_in->c( 10 ) ).
lo_specification->add_type( 'TT_PRODUCTS' )->for_table_type(
  io_row_type = lo_specification->own->type( 'TV_PRODUCT' )
  io_access   = xco_cp_table_type=>access->standard_table
).
lo_specification->add_type( 'TV_ID' )->for( xco_cp_abap=>type-built_in-
>c( 10 ) ).
lo_specification->add_data( 'ID' )->set_type( lo_specification->own-
>type( 'TV_ID' )
  )->set_read_only( ).
DATA(lo_get_favorite_products) = lo_specification-
>add_method( 'GET_FAVORITE_PRODUCTS' ).
lo_get_favorite_products->add_returning_parameter( 'RT_FAVORITE_PRODUCTS'
  )->set_type( lo_specification->own->type( 'TT_PRODUCTS' ) ).
ENDMETHOD.
METHOD add_clas.
DATA(lo_specification) = io_put_operation->for-clas-
>add_object( co_class_name
  )->set_package( co_package
  )->create_form_specification( ).
lo_specification->set_short_description( 'Customer implementation' ).
lo_specification->definition-
>set_create_visibility( xco_cp_abap_objects=>visibility->private ).
lo_specification->definition->add_interface( co_interface_name ).
lo_specification->definition->section-public->add_alias( 'TV_ID'
  )->set_component( 'TV_ID'
  )->set_interface( co_interface_name ).
" CREATE factory method.
DATA(lo_create) = lo_specification->definition->section-public-
>add_class_method( 'CREATE' ).
lo_create->add_importing_parameter( 'IV_ID' )->set_type( lo_specification-
>own->type( 'TV_ID' ) ).
lo_create->add_returning_parameter( 'RO_INSTANCE' )-
>set_type( xco_cp_abap=>interface( co_interface_name ) ).
```

```

lo_specification->implementation->add_method( 'CREATE' )-
>set_source( VALUE #(
    ( |ro_instance = new { co_class_name }( iv_id ).| )
) ).  

" CONSTRUCTOR.  

DATA(lo_constructor) = lo_specification->definition->section-private-
>add_method( 'CONSTRUCTOR' ).  

lo_constructor->add_importing_parameter( 'IV_ID' )-
>set_type( lo_specification->own->type( 'TV_ID' ) ).  

lo_specification->implementation->add_method( 'CONSTRUCTOR' )-
>set_source( VALUE #(
    ( |{ co_interface_name }~id = iv_id.| )
) ).  

" Implementation of GET_FAVORITE_PRODUCTS interface method.  

lo_specification->implementation->add_method( |
{ co_interface_name }~GET_FAVORITE_PRODUCTS|
    )->set_source( VALUE #(
        ( |" Source code goes here...| )
    ) ).  

ENDMETHOD.  

ENDCLASS.

```

Note that the XCO Generation APIs for classes also allow the specification of local types and local test classes:

- Method `ADD_LOCAL_CLASS` on `IF_XCO_CP_GEN_CLAS_S_FORM` can be used to add a local type to the generated class. The content of the local class can be fully specified via the returned local class specification `IF_XCO_GEN_CLAS_S_FO_LCL_CLASS`
- Method `ADD_TEST_CLASS` on `IF_XCO_CP_GEN_CLAS_S_FORM` can be used to add a local test class to the generated class. The content of the local test class can be fully specified via the returned test class specification `IF_XCO_GEN_CLAS_S_FO_TST_CLASS`. Note that the API `XCO_CP_ABAP_UNIT` offers XCO enumeration constants for ABAP Unit durations and risk levels

### 4.2.11.16.3.3.7 Patching a Message Class

The following code sample illustrates how PATCH operations can be used to change parts of an existing messages class:

#### ↔ Sample Code

```

"! <p class="shorttext synchronized" lang="EN">
"!   XCO Documentation CP: Generation PATCH
"! </p>
"!
"! Code sample for executing PATCH operations for messages classes (MSAGs).
"!
"! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
"! with an existing package and a modifiable Workbench transport matching the
transport
"! target of the package.
CLASS zcl_xco_doc_cp_cs_gen_patch DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PUBLIC SECTION.
  METHODS:
    constructor.

  PROTECTED SECTION.
  METHODS:

```

```

main REDEFINITION.

PRIVATE SECTION.
CONSTANTS:
  co_package          TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
  co_transport        TYPE sxco_transport VALUE 'X08K900164',
  co_message_class_name TYPE sxco_mc_object_name VALUE
'ZXCO_DEMO_MSG_CLS'.

DATA:
  mo_environment TYPE REF TO if_xco_cp_gen_env_dev_system.

METHODS:
  put_msag,
  patch_msag_1,
  patch_msag_2,
  read_msag
    IMPORTING
      out TYPE REF TO if_xco_adt_classrun_out.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_gen_patch IMPLEMENTATION.
METHOD constructor.
  super->constructor( ).

  mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.

METHOD main.
  put_msag( ).
  read_msag( out ).

  patch_msag_1( ).
  read_msag( out ).

  patch_msag_2( ).
  read_msag( out ).
ENDMETHOD.

METHOD put_msag.
  DATA(lo_put_operation) = mo_environment->for-msag-
>create_put_operation( ).

  DATA(lo_specification) = lo_put_operation-
>add_object( co_message_class_name
    )->set_package( co_package
    )->create_form_specification( ).
  lo_specification->set_short_description( 'Generated message class' ).

  lo_specification->add_message( '000'
    )->set_short_text( 'First message' ).

  lo_put_operation->execute( ).
ENDMETHOD.

METHOD patch_msag_1.
  DATA(lo_patch_operation) = mo_environment->for-msag-
>create_patch_operation( ).

  DATA(lo_object) = lo_patch_operation->add_object( co_message_class_name ).

  " Update existing message.

```

```

        lo_object->for-update->message( '000' )->set_short_text( 'First message
(UPDATED)' ).

        " Insert new message.
        lo_object->for-insert->add_message( '001' )->set_short_text( 'Second
message' ).

        " Modify non-existing message (will insert the message).
        lo_object->for-modify->add_message( '002' )->set_short_text( 'Third
message' ).

        lo_patch_operation->execute( ).
ENDMETHOD.

METHOD patch_msag_2.
    DATA(lo_patch_operation) = mo_environment->for-msag-
>create_patch_operation( ).

    DATA(lo_object) = lo_patch_operation->add_object( co_message_class_name ).

    " Modify non-existing message (will update the message).
    lo_object->for-modify->add_message( '002' )->set_short_text( 'Third
message (UPDATED)' ).

    " Delete message
    lo_object->for-delete->add_message( '000' ).

    lo_patch_operation->execute( ).
ENDMETHOD.

METHOD read_msag.
    DATA(lo_message_class) = xco_cp_abap_repository=>object->msag-
>for( co_message_class_name ).

    out->write( |Messages of message class { lo_message_class->name } :| ).

    LOOP AT lo_message_class->messages->all->get( ) INTO DATA(lo_message).
        out->write( |Message { lo_message->number } :| ).

        DATA(ls_message) = lo_message->content( )->get( ).
        out->write( ls_message ).

    ENDLOOP.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.16.3.3.8 Patching Methods of Interfaces and Classes

The XCO Generation APIs allow to incrementally modify existing interfaces and classes through PATCH operations.

The following two code samples illustrate how methods of interfaces and classes can be patched. In each code sample, the MAIN method will execute one of four phases depending on the current state of the interface or class. To see each phase in action simply execute each class several times in a row.

Here is how it looks for interfaces:

### ↔ Sample Code

```

"! Code sample for patching methods of an interface.
"!

```

```

    ! IMPORTANT: The value of the constant CO_PACKAGE must be replaced with an
existing
    ! development package.
CLASS zcl_xco_doc_cp_cs_patch_in_me DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.

PROTECTED SECTION.
METHODS:
    main REDEFINITION.

PRIVATE SECTION.
CONSTANTS:
    co_package      TYPE sxco_package VALUE 'ZMY_PACKAGE',
    co_interface_name TYPE sxco_ao_object_name VALUE
'ZIF_XCO_DMO_GEN_INTF_PATCH_MTD',
    co_first_method  TYPE sxco_ao_component_name VALUE 'FIRST_METHOD',
    co_second_method TYPE sxco_ao_component_name VALUE 'SECOND_METHOD',
    co_third_method  TYPE sxco_ao_component_name VALUE 'THIRD_METHOD'.

DATA:
    mo_interface TYPE REF TO if_xco_ao_interface.

METHODS:
    phase_1
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_2
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_3
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_4
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    get_environment
        IMPORTING
            out          TYPE REF TO if_xco_adt_classrun_out
        RETURNING
            VALUE(ro_environment) TYPE REF TO if_xco_cp_gen_env_dev_system,
    get_transport_request
        IMPORTING
            out          TYPE REF TO if_xco_adt_classrun_out
        RETURNING
            VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_patch_in_me IMPLEMENTATION.
METHOD constructor.
    super->constructor( ).

    mo_interface = xco_cp_abap=>interface( co_interface_name ).
ENDMETHOD.

METHOD main.
    IF mo_interface->exists( ) EQ abap_false.
        " In the first phase the interface is created via a PUT operation.
        phase_1( out ).
```

```

        out->write( |Executed phase 1| ).

ELSEIF mo_interface->component->method( 'FIRST_METHOD' )->exists( ) EQ
abap_false.
    " In the second phase two new methods are inserted into the interface
    " via a single PATCH operation.
    phase_2( out ).

    out->write( |Executed phase 2| ).

ELSEIF mo_interface->component->method( 'SECOND_METHOD' )->exists( ) EQ
abap_true.
    " In the third phase one method is deleted, a new method is inserted and
    " an existing method is updated via a single PATCH operation.
    phase_3( out ).

    out->write( |Executed phase 3| ).

ELSE.
    " In the fourth phase the interface is deleted via a DELETE operation.
    phase_4( out ).

    out->write( |Executed phase 4| ).

ENDIF.

ENDMETHOD.

METHOD phase_1.
    DATA(lo_put_operation) = get_environment( out )->create_put_operation( ).

    DATA(lo_specification) = lo_put_operation->for-intf-
>add_object( mo_interface->name
    )->set_package( co_package
    )->create_form_specification( ).
    lo_specification->set_short_description( 'My interface' ).

    " Data
    lo_specification->add_data( 'ID'
    )->set_type( xco_cp_abap=>type-built_in->i
    )->set_read_only( ).

    lo_put_operation->execute( ).

ENDMETHOD.

METHOD phase_2.
    DATA(lo_patch_operation) = get_environment( out )->for-intf-
>create_patch_operation( ).

    DATA(lo_patch_operation_object) = lo_patch_operation-
>add_object( mo_interface->name ).

    " Insert first method
    DATA(lo_first_method) = lo_patch_operation_object->for-insert-
>add_method( co_first_method
    )->set_default_ignore( ).

    lo_first_method->add_importing_parameter( 'IMP_1'
    )->set_type( xco_cp_abap=>type-built_in->string ).
    lo_first_method->add_importing_parameter( 'IMP_2'
    )->set_type( xco_cp_abap=>type-built_in->string ).
    lo_first_method->add_changing_parameter( 'CHA_1'
    )->set_type( xco_cp_abap=>interface( 'IF_XCO_NEWS' ) ).

    lo_first_method->add_exception( 'CX_ABAP_INVALID_VALUE' ).

    " Insert second method
    DATA(lo_second_method) = lo_patch_operation_object->for-insert-
>add_method( co_second_method
    )->set_default_fail( ).

    lo_patch_operation->execute( ).

```

```

ENDMETHOD.

METHOD phase_3.
    DATA(lo_patch_operation) = get_environment( out )->for-intf-
>create_patch_operation( ).

    DATA(lo_patch_operation_object) = lo_patch_operation-
>add_object( mo_interface->name ).

    " Update first method.
    DATA(lo_first_method) = lo_patch_operation_object->for-update-
>add_method( co_first_method
        )->set_default_fail( ).

    lo_first_method->for-update->add_importing_parameter( 'IMP_1'
        )->set_pass_by_value(
        )->set_optional( ).

    lo_first_method->for-delete->add_importing_parameter( 'IMP_2' ).

    lo_first_method->for-insert->add_importing_parameter( 'IMP_3'
        )-
>set_type( xco_cp_abap_dictionary=>data_element( 'SXCO_AO_OBJECT_NAME' ) ).

    lo_first_method->for-update->add_exception( 'CX_ABAP_INVALID_VALUE'
        )->set_resumable( ).

    " Delete second method.
    lo_patch_operation_object->for-delete->add_method( co_second_method ).

    " Insert third method.
    lo_patch_operation_object->for-insert->add_method( co_third_method
        )->set_default_fail( ).

    lo_patch_operation->execute( ).
ENDMETHOD.

METHOD phase_4.
    DATA(lo_delete_operation) = get_environment( out )->for-intf-
>create_delete_operation( ).
    lo_delete_operation->add_object( mo_interface->name ).
    lo_delete_operation->execute( ).
ENDMETHOD.

METHOD get_environment.
    DATA(lo_transport_request) = get_transport_request( out ).

    ro_environment = xco_cp_generation=>environment-
>dev_system( lo_transport_request->value ).
ENDMETHOD.

METHOD get_transport_request.
    DATA(lo_lock) = mo_interface->if_xco_cts_changeable~get_object(
        )->get_lock( ).

    IF lo_lock->exists( ) EQ abap_true.
        DATA(lv_transport) = lo_lock->get_transport( ).

        ro_transport_request = xco_cp_cts=>transport->for( lv_transport
            )->get_request( ).

        out->write( |Using transport request { ro_transport_request->value } |
(Interface is already locked).| ).
    ELSE.
        DATA(lo_transport_target) = xco_cp_abap_repository=>package-
>for( co_package
            )->read(
            )->property-transport_layer->get_transport_target( ).
    ENDIF.
ENDMETHOD.

```

```

        ro_transport_request = xco_cp_cts->transports-
>workbench( lo_transport_target->value
        )->create_request( 'Generated transport request' ).

        out->write( |Using newly generated transport request
{ ro_transport_request->value }| ).

        ENDIF.
    ENDMETHOD.
ENDCLASS.
```

and here is how it looks for classes:

### ↳ Sample Code

```

! Code sample for patching methods of a class.
!
! IMPORTANT: The value of the constant CO_PACKAGE must be replaced with an
existing
! development package.
CLASS zcl_xco_doc_cp_cs_patch_cl_me DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.
PROTECTED SECTION.
METHODS:
    main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:
    co_package      TYPE sxco_package VALUE 'ZMY_PACKAGE',
    co_class_name   TYPE sxco_ao_object_name VALUE
'ZCL_XCO_DMO_GEN_CLAS_PATCH_MTD',
    co_first_method TYPE sxco_clas_method_name VALUE 'FIRST_METHOD',
    co_second_method TYPE sxco_clas_method_name VALUE 'SECOND_METHOD',
    co_third_method TYPE sxco_clas_method_name VALUE 'THIRD_METHOD'.
DATA:
    mo_class TYPE REF TO if_xco_ao_class.
METHODS:
    phase_1
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_2
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_3
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    phase_4
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
get_environment
    IMPORTING
        out
            TYPE REF TO if_xco_adt_classrun_out
    RETURNING
        VALUE(ro_environment) TYPE REF TO if_xco_cp_gen_env_dev_system,
get_transport_request
    IMPORTING
        out
            TYPE REF TO if_xco_adt_classrun_out
    RETURNING
        VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_patch_cl_me IMPLEMENTATION.
METHOD constructor.
    super->constructor( ).
    mo_class = xco_cp_abap=>class( co_class_name ).
```

```

ENDMETHOD.
METHOD main.
  IF mo_class->exists( ) EQ abap_false.
    " In the first phase the class is created via a PUT operation.
    phase_1( out ).
    out->write( |Executed phase 1| ).
  ELSEIF mo_class->definition->section-public->component-
>method( 'FIRST_METHOD' )->exists( ) EQ abap_false.
    " In the second phase two new methods are inserted into the class, one
    in the public section
    " and one in the private section, via a single PATCH operation.
    phase_2( out ).
    out->write( |Executed phase 2| ).
  ELSEIF mo_class->definition->section-private->component-
>method( 'SECOND_METHOD' )->exists( ) EQ abap_true.
    " In the third phase one method is deleted, a new method is inserted
    into the protected
    " section and an existing method is updated via a single PATCH
    operation.
    phase_3( out ).
    out->write( |Executed phase 3| ).
  ELSE.
    " In the fourth phase the class is deleted via a DELETE operation.
    phase_4( out ).
    out->write( |Executed phase 4| ).
  ENDIF.
ENDMETHOD.
METHOD phase_1.
  DATA(lo_put_operation) = get_environment( out )->create_put_operation( ).
  DATA(lo_specification) = lo_put_operation->for-clas->add_object( mo_class-
>name
  )->set_package( co_package
  )->create_form_specification( ).
  lo_specification->set_short_description( 'My class' ).
  " Data
  lo_specification->definition->section-public->add_data( 'ID'
  )->set_type( xco_cp_abap=>type-built_in->i
  )->set_read_only( ).
  lo_put_operation->execute( ).
ENDMETHOD.
METHOD phase_2.
  DATA(lo_patch_operation) = get_environment( out )->for-clas-
>create_patch_operation( ).
  DATA(lo_patch_operation_object) = lo_patch_operation-
>add_object( mo_class->name ).
  " First we specify the definition part of the class.
  " Insert the first method into the public section of the class.
  DATA(lo_first_method) = lo_patch_operation_object->for-insert->definition-
>section-public->add_method( co_first_method ).
  lo_first_method->add_importing_parameter( 'IMP_1'
  )->set_type( xco_cp_abap=>type-built_in->string ).
  lo_first_method->add_importing_parameter( 'IMP_2'
  )->set_type( xco_cp_abap=>type-built_in->string ).
  lo_first_method->add_changing_parameter( 'CHA_1'
  )->set_type( xco_cp_abap=>interface( 'IF_XCO_NEWS' ) ).
  lo_first_method->add_exception( 'CX_ABAP_INVALID_VALUE' ).
  " Insert the second method into the private section of the class.
  DATA(lo_second_method) = lo_patch_operation_object->for-insert-
>definition->section-private->add_method( co_second_method ).
  " After the definition has been specified we also specify the
  implementation part. If no implementation
  " is specified for an inserted method it will have an empty
  implementation (as is the case for
  " the second method).
  lo_patch_operation_object->for-insert->implementation-
>add_method( co_first_method
  )->set_source( VALUE #(
    ` " Source code goes here` )

```

```

        ) .
    lo_patch_operation->execute( ) .
ENDMETHOD.

METHOD phase_3.
    DATA(lo_patch_operation) = get_environment( out )->for-class-
>create_patch_operation( ) .
    DATA(lo_patch_operation_object) = lo_patch_operation-
>add_object( mo_class->name ) .
    " As in phase 2, we first specify the definition part of the class.
    " Update first method.
    DATA(lo_first_method) = lo_patch_operation_object->for-update->definition-
>section-public->add_method( co_first_method ) .
    lo_first_method->for-update->add_importing_parameter( 'IMP_1'
        )->set_pass_by_value(
        )->set_optional( ) .
    lo_first_method->for-delete->add_importing_parameter( 'IMP_2' ) .
    lo_first_method->for-insert->add_importing_parameter( 'IMP_3'
        )-
>set_type( xco_cp_abap_dictionary=>data_element( 'SXCO_AO_OBJECT_NAME' ) ) .
    lo_first_method->for-update->add_exception( 'CX_ABAP_INVALID_VALUE'
        )->set_resumable( ) .
    " Delete second method.
    lo_patch_operation_object->for-delete->definition->section-private-
>add_method( co_second_method ) .
    " Insert the third method into the protected section of the class.
    lo_patch_operation_object->for-insert->definition->section-protected-
>add_method( co_third_method ) .
    " Now the implementation part of the class is specified.
    lo_patch_operation_object->for-update->implementation-
>add_method( co_first_method
        )->set_source( VALUE #(
            ( ` RAISE EXCEPTION TYPE cx_abap_invalid_value.` )
        ) ) .
    lo_patch_operation_object->for-insert->implementation-
>add_method( co_third_method
        )->set_source( VALUE #(
            ( ` " Third method implementation` )
        ) ) .
    lo_patch_operation->execute( ) .
ENDMETHOD.

METHOD phase_4.
    DATA(lo_delete_operation) = get_environment( out )->for-class-
>create_delete_operation( ) .
    lo_delete_operation->add_object( mo_class->name ) .
    lo_delete_operation->execute( ) .
ENDMETHOD.

METHOD get_environment.
    DATA(lo_transport_request) = get_transport_request( out ) .
    ro_environment = xco_cp_generation=>environment-
>dev_system( lo_transport_request->value ) .
ENDMETHOD.

METHOD get_transport_request.
    DATA(lo_lock) = mo_class->if_xco_cts_changeable~get_object(
        )->get_lock( ) .
    IF lo_lock->exists( ) EQ abap_true.
        DATA(lv_transport) = lo_lock->get_transport( ) .
        ro_transport_request = xco_cp_cts=>transport->for( lv_transport
            )->get_request( ) .
        out->write( |Using transport request { ro_transport_request->value } |
(Class is already locked).| ) .
    ELSE.
        DATA(lo_transport_target) = xco_cp_abap_repository=>package-
>for( co_package
        )->read(
        )-property-transport_layer->get_transport_target( ) .
        ro_transport_request = xco_cp_cts=>transports-
>workbench( lo_transport_target->value
        )->create_request( 'Generated transport request' ) .

```

```

        out->write( |Using newly generated transport request
{ ro_transport_request->value }.| ).
ENDIF.
ENDMETHOD.
ENDCLASS.
```

### 4.2.11.16.3.3.9 Patching Attributes of Interfaces and Classes

The code samples illustrate how the interface and class Read APIs can be combined with PATCH operations to conditionally change attributes of interfaces and classes.

In the first example an attribute is inserted into an interface if it's not yet part of it and deleted if it's already a part of the interface. By running the class several times in a row the attribute will be continuously inserted and deleted.

#### ↳ Sample Code

```

! Code sample for using the interface Read APIs together with the INTF PATCH
operation
! functionality to "toggle" a DATA attribute on an interface.
!
! IMPORTANT: The interface referred to by the constant CO_INTERFACE_NAME
needs to
! exist prior to executing this code sample.
CLASS zcl_xco_doc_cp_cs_chg_intf DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.

PROTECTED SECTION.
METHODS:
    main REDEFINITION.

PRIVATE SECTION.
CONSTANTS:
    co_interface_name TYPE sxco_ao_object_name VALUE 'ZIF_INTERFACE',
    co_numeric_id      TYPE sxco_ao_component_name VALUE 'NUMERIC_ID'.

METHODS:
    get_transport_request
        IMPORTING
            io_interface          TYPE REF TO if_xco_ao_interface
            out                  TYPE REF TO if_xco_adt_classrun_out
        RETURNING
            VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_chg_intf IMPLEMENTATION.
METHOD constructor.
    super->constructor( ).
```

```

        out      = out
).

DATA(lo_environment) = xco_cp_generation->environment-
>dev_system( lo_transport_request->value ).

DATA(lo_patch_operation) = lo_environment->for-intf-
>create_patch_operation( ).
DATA(lo_object) = lo_patch_operation->add_object( co_interface_name ).

IF lo_interface->component->data( co_numeric_id )->exists( ) EQ
abap_false.
    lo_object->for-insert->add_data( co_numeric_id
    )->set_type( xco_cp_abap=>type-built_in->int8
    )->set_read_only( ).

    out->write( |Inserting DATA { co_numeric_id }| ).

ELSE.
    lo_object->for-delete->add_data( co_numeric_id ).

    out->write( |Deleting DATA { co_numeric_id }| ).

ENDIF.

lo_patch_operation->execute( ).

ENDMETHOD.

METHOD get_transport_request.
DATA(lo_lock) = io_interface->if_xco_cts_changeable~get_object(
    )->get_lock( ).

IF lo_lock->exists( ) EQ abap_true.
    DATA(lv_transport) = lo_lock->get_transport( ).

    ro_transport_request = xco_cp_cts->transport->for( lv_transport
    )->get_request( ).

    out->write( |Using transport request { ro_transport_request->value }|
(Interface is already locked).| ).

ELSE.
    DATA(lo_transport_target) = io_interface->if_xco_ar_object~get_package(
        )->read(
        )-property-transport_layer->get_transport_target( ).

    ro_transport_request = xco_cp_cts->transports-
>workbench( lo_transport_target->value
    )->create_request( 'Generated transport request' ).

    out->write( |Using newly generated transport request
{ ro_transport_request->value }.| ).

ENDIF.
ENDMETHOD.
ENDCLASS.
```

The second example "moves" an attribute through all three sections of a class. By running the below class several times in a row the attribute will first move from the public section to the protected section, then to the private section and then back to the public section to repeat the cycle:

#### ↔ Sample Code

```

! Code sample for using the class Read APIs together with the CLAS PATCH
operation
! functionality to "move" a DATA attribute across the sections of the class.
!
! IMPORTANT: The class referred to by the constant CO_CLASS_NAME needs to
! exist prior to executing this code sample.
```

```

CLASS zcl_xco_doc_cp_cs_chg_clas DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
  constructor.
PROTECTED SECTION.
METHODS:
  main REDEFINITION.
PRIVATE SECTION.
TYPES:
  BEGIN OF ts_section,
    name          TYPE string,
    class_definition TYPE REF TO if_xco_clas_definition_section,
    for_delete    TYPE REF TO if_xco_gen_clas_s_de_d_section,
    for_insert    TYPE REF TO if_xco_gen_clas_s_in_d_section,
  END OF ts_section,
  tt_section TYPE STANDARD TABLE OF ts_section WITH EMPTY KEY.
CONSTANTS:
  co_class_name TYPE sxco_ao_object_name VALUE 'ZCL_CLASS',
  co_numeric_id TYPE sxco_ao_component_name VALUE 'NUMERIC_ID'.
DATA:
  mo_class TYPE REF TO if_xco_ao_class.
METHODS:
  get_generation_environment
    IMPORTING
      out
    RETURNING
      VALUE(ro_generation_environment) TYPE REF TO
if_xco_cp_gen_env_dev_system,
  get_transport_request
    IMPORTING
      out
    RETURNING
      VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request,
get_sections
  IMPORTING
    io_object      TYPE REF TO if_xco_cp_gen_clas_d_o_pat_obj
  RETURNING
    VALUE(rt_sections) TYPE tt_section.
ENDCLASS.
CLASS zcl_xco_doc_cp_cs_chg_clas IMPLEMENTATION.
METHOD constructor.
  super->constructor( ).
  mo_class = xco_cp_abap=>class( co_class_name ).
ENDMETHOD.
METHOD main.
  DATA(lo_patch_operation) = get_generation_environment( out )->for-clas-
>create_patch_operation( ).
  DATA(lo_object) = lo_patch_operation->add_object( co_class_name ).
  DATA(lt_sections) = get_sections( lo_object ).
  LOOP AT lt_sections INTO DATA(ls_section).
    DATA(lv_current_section) = sy-tabix.
    IF ls_section-class_definition->component->data( co_numeric_id )-
>exists( ) EQ abap_true.
      ls_section-for_delete->add_data( co_numeric_id ).
      out->write( |Deleting DATA { co_numeric_id } from section
{ ls_section-name }.| ).
      EXIT.
    ENDIF.
  ENDLOOP.
  " Insert attribute to next section.
  DATA(ls_next_section) = lt_sections[ lv_current_section MOD 3 + 1 ].
  ls_next_section-for_insert->add_data( co_numeric_id
    )->set_type( xco_cp_abap=>type-built_in->int8 ).
  out->write( |Inserting DATA { co_numeric_id } into section
{ ls_next_section-name }.| ).
  lo_patch_operation->execute( ).
ENDMETHOD.

```

```

METHOD get_generation_environment.
  DATA(lo_transport_request) = get_transport_request( out ).
  ro_generation_environment = xco_cp_generation->environment-
>dev_system( lo_transport_request->value ).
ENDMETHOD.

METHOD get_transport_request.
  DATA(lo_lock) = mo_class->if_xco_cts_changeable~get_object(
    )->get_lock( ).
  IF lo_lock->exists( ) EQ abap_true.
    DATA(lv_transport) = lo_lock->get_transport( ).
    ro_transport_request = xco_cp_cts->transport->for( lv_transport
    )->get_request( ).
    out->write( |Using transport request { ro_transport_request->value }|
(Class is already locked).| ).
  ELSE.
    DATA(lo_transport_target) = mo_class->if_xco_ar_object~get_package(
      )->read(
      )-property-transport_layer->get_transport_target( ).
    ro_transport_request = xco_cp_cts->transports-
>workbench( lo_transport_target->value
      )->create_request( 'Generated transport request' ).
    out->write( |Using newly generated transport request
{ ro_transport_request->value }.| ).
  ENDIF.
ENDMETHOD.

METHOD get_sections.
  DATA(ls_public_section) = VALUE ts_section(
    name          = `PUBLIC`
    class_definition = mo_class->definition->section-public
    for_delete     = io_object->for-delete->definition->section-public
    for_insert     = io_object->for-insert->definition->section-public
  ).
  APPEND ls_public_section TO rt_sections.
  DATA(ls_protected_section) = VALUE ts_section(
    name          = `PROTECTED`
    class_definition = mo_class->definition->section-protected
    for_delete     = io_object->for-delete->definition->section-protected
    for_insert     = io_object->for-insert->definition->section-protected
  ).
  APPEND ls_protected_section TO rt_sections.
  DATA(ls_private_section) = VALUE ts_section(
    name          = `PRIVATE`
    class_definition = mo_class->definition->section-private
    for_delete     = io_object->for-delete->definition->section-private
    for_insert     = io_object->for-insert->definition->section-private
  ).
  APPEND ls_private_section TO rt_sections.
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.16.3.3.10 Function Module Generation

The following code sample illustrates the generation of function modules:

### ↔ Sample Code

```

! Code sample for generating a function module along with a function group
! that contains it.
!
! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced

```

```

    ! with an existing package and a modifiable Workbench transport request
    ! matching the
    ! transport target of the package.
CLASS zcl_xco_doc_cp_cs_gen_fm DEFINITION PUBLIC FINAL
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
PUBLIC SECTION.
METHODS:
    constructor.
PROTECTED SECTION.
METHODS:
    main REDEFINITION.
PRIVATE SECTION.
CONSTANTS:
    co_package           TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
    co_transport         TYPE sxco_transport VALUE 'X08K900164',
    co_function_group_name  TYPE sxco_fg_object_name VALUE
'ZXCO_GEN_FUGR_FG',
    co_function_module_name TYPE sxco_fm_name VALUE 'ZXCO_GEN_FUGR_FM'.
DATA:
    mo_function_group   TYPE REF TO if_xco_function_group,
    mo_function_module  TYPE REF TO if_xco_function_module,
    mo_environment       TYPE REF TO if_xco_cp_gen_env_dev_system.
METHODS:
    set_up_function_group
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    post_function_group,
    delete_function_modules
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out,
    insert_function_modules
        IMPORTING
            out TYPE REF TO if_xco_adt_classrun_out.
ENDCLASS.
CLASS zcl_xco_doc_cp_cs_gen_fm IMPLEMENTATION.
METHOD constructor.
    super->constructor( ).
    mo_function_group = xco_cp_abap=>function_group( co_function_group_name ).
    mo_function_module =
xco_cp_abap=>function_module( co_function_module_name ).
    mo_environment = xco_cp_generation=>environment-
>dev_system( co_transport ).
ENDMETHOD.
METHOD main.
    set_up_function_group( out ).
    delete_function_modules( out ).
    insert_function_modules( out ).
ENDMETHOD.
METHOD set_up_function_group.
    IF mo_function_group->if_xco_ar_object~exists( ) EQ abap_true.
        out->write( |Function group { mo_function_group->name } already
exists.| ).
    ELSE.
        post_function_group( ).
        out->write( |Function group { mo_function_group->name } created
successfully.| ).
    ENDIF.
ENDMETHOD.
METHOD post_function_group.
    DATA(lo_post_operation) = mo_environment->for-fugr-
>create_post_operation( ).
    DATA(lo_form_specification) = lo_post_operation-
>add_object( xco_cp_name=>choice->fixed( co_function_group_name )
    )->set_package( co_package
    )->create_form_specification( ).
    lo_form_specification->set_short_description( 'Generated FUGR' ).
    lo_post_operation->execute( ).
ENDMETHOD.

```

```

METHOD delete_function_modules.
  DATA(lo_patch_operation) = mo_environment->for-fugr-
>create_patch_operation( ).
  DATA(lt_function_module_names) = mo_function_group->function_modules->all-
>get_names( ).
  LOOP AT lt_function_module_names INTO DATA(lv_function_module_name).
    lo_patch_operation->add_object( co_function_group_name
      )->for-delete->add_function_module( lv_function_module_name ).
  ENDLOOP.
  lo_patch_operation->execute( ).
  out->write( |Function modules of function group { mo_function_group-
>name } deleted successfully.| ).
ENDMETHOD.

METHOD insert_function_modules.
  DATA(lo_patch_operation) = mo_environment->for-fugr-
>create_patch_operation( ).
  DATA(lo_function_module) = lo_patch_operation-
>add_object( co_function_group_name
    )->for-insert->add_function_module( co_function_module_name ).
  " IMPORTING parameters.
  lo_function_module->add_importing_parameter( 'IM_CLASS'
    )->set_type( xco_cp_abap=>class( 'CL_XCO_AD_BUILT_IN_TYPE' )
    )->set_optional( ).
  lo_function_module->add_importing_parameter( 'IM_INTERFACE'
    )->set_type( xco_cp_abap=>interface( 'IF_XCO_NEWS' ) ).
  lo_function_module->add_importing_parameter( 'IM_ABAP_BUILT_IN_TYPE'
    )->set_type( xco_cp_abap=>type-built_in->i
    )->set_default_value( '5' ).
  lo_function_module->add_importing_parameter( 'IM_ABAP_BIT_REFERENCE'
    )->set_type( xco_cp_abap=>type-built_in->i->reference( ) ).
  lo_function_module->add_importing_parameter( 'IM_ABAP_GENERIC_TYPE'
    )->set_type( xco_cp_abap=>type-generic->c ).
  lo_function_module->add_importing_parameter( 'IM_ABAP_GT_REFERENCE'
    )->set_type( xco_cp_abap=>type-generic->data->reference( ) ).
  lo_function_module->add_importing_parameter( 'IM_DATA_ELEMENT'
    )->set_type( xco_cp_abap_dictionary=>data_element( 'SXCO_FM_NAME' ) ).
  " EXPORTING parameters.
  lo_function_module->add_exporting_parameter( 'EX_STRING'
    )->set_type( xco_cp_abap=>type-built_in->string ).
  " CHANGING parameters.
  lo_function_module->add_exporting_parameter( 'CH_ANY'
    )->set_type( xco_cp_abap=>type-generic->any ).
  " Exceptions.
  lo_function_module->add_exception( 'CX_ABAP_INVALID_NAME' ).
  lo_function_module->add_exception( 'CX_ABAP_INVALID_VALUE'
    )->set_resumable( abap_true ).
  " Source code.
  lo_function_module->set_source_code( xco_cp=>strings( VALUE #(
    ( |DATA lo_string TYPE REF TO if_xco_string.| )
  ) ) ).
  lo_patch_operation->execute( ).
  out->write( |Inserted function module { mo_function_module->name } into
function group { mo_function_group->name }.| ).
ENDMETHOD.

ENDCLASS.

```

## 4.2.11.16.3.3.11 Updating the Source Code of a Function Module

The following code sample illustrates the source code of an already existing function module that can be updated:

## ↔ Sample Code

```
! Code sample for updating the source code of an existing function module.  
!  
! IMPORTANT: The function module referred to by the constant  
CO_FUNCTION_MODULE_NAME needs to  
! exist prior to executing this code sample and must not already be locked  
on a transport  
! request.  
CLASS zcl_xco_doc_cp_cs_chg_fm DEFINITION PUBLIC FINAL  
    INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.  
PUBLIC SECTION.  
METHODS:  
    constructor.  
  
PROTECTED SECTION.  
METHODS:  
    main REDEFINITION.  
  
PRIVATE SECTION.  
CONSTANTS:  
    co_function_module_name TYPE sxco_fm_name VALUE 'ZMY_FUNCTION_MODULE'.  
  
DATA:  
    mo_function_module TYPE REF TO if_xco_function_module.  
  
METHODS:  
    generate_transport_request  
        IMPORTING  
            out  
        RETURNING  
            VALUE(ro_transport_request) TYPE REF TO if_xco_cp_tr_request,  
  
    release_transport_request  
        IMPORTING  
            io_transport_request TYPE REF TO if_xco_cp_tr_request  
            out  
        RETURNING  
            TYPE REF TO if_xco_adt_classrun_out.  
ENDCLASS.  
  
CLASS zcl_xco_doc_cp_cs_chg_fm IMPLEMENTATION.  
METHOD constructor.  
    super->constructor( ).  
  
    mo_function_module =  
xco_cp_abap->function_module( co_function_module_name ).  
ENDMETHOD.  
  
METHOD main.  
    DATA(lo_transport_request) = generate_transport_request( out ).  
  
    DATA(lo_patch_operation) = xco_cp_generation->environment-  
>dev_system( lo_transport_request->value  
        )->for-fugr->create_patch_operation( ).  
  
    DATA(lo_function_group) = mo_function_module->get_function_group( ).  
    lo_patch_operation->add_object( lo_function_group->name  
        )->for-update->add_function_module( mo_function_module->name  
        )->set_source_code( xco_cp=>strings( VALUE #(  
            ( ` " New source code goes here...` )  
        ) ) ).  
  
    lo_patch_operation->execute( ).  
  
    release_transport_request(  
        io_transport_request = lo_transport_request  
        out  
        = out  
    ).
```

```

ENDMETHOD.

METHOD generate_transport_request.
  DATA(lo_transport_target) = mo_function_module->get_function_group(
    )->if_xco_ar_object~get_package(
    )->read(
    )-property-transport_layer->get_transport_target( ).

  ro_transport_request = xco_cp_cts=>transports-
>workbench( lo_transport_target->value
  )->create_request( 'Generated transport request' ).

  out->write( |Using newly generated transport request
{ ro_transport_request->value }| ).
ENDMETHOD.

METHOD release_transport_request.
  DATA(lt_tasks) = io_transport_request->get_tasks( ).

  LOOP AT lt_tasks INTO DATA(lo_task).
    lo_task->release( ).
  ENDLOOP.

  io_transport_request->release( ).

  out->write( |Transport request { io_transport_request->value } has been
released.| ).
ENDMETHOD.
ENDCLASS.

```

## 4.2.11.16.3.3.12 Table Functions and AMDP

The code sample below provides a self-contained example for the generation of a CDS table function. The CDS table function is generated together with the class providing the AMDP implementation and the underlying database table in a single PUT operation.

### ↔ Sample Code

```

! Code sample for generating a CDS table function together with an AMDP class
! and database table.
!
#! IMPORTANT: The values of the constants CO_PACKAGE and CO_TRANSPORT must be
replaced
#! with an existing package and a modifiable Workbench transport request
matching the
#! transport target of the package.
CLASS zcl_xco_doc_cp_cs_gen_tbl_fnc DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PROTECTED SECTION.
  METHODS:
    main REDEFINITION.
  PRIVATE SECTION.
  CONSTANTS:
    co_package          TYPE sxco_package VALUE 'ZXCO_CP_PACKAGE',
    co_transport        TYPE sxco_transport VALUE 'X08K900164',
    co_database_table_name TYPE sxco_dbt_object_name VALUE 'ZXCO_TF_DBT',
    co_table_function_name TYPE sxco_cds_object_name VALUE
'ZXCO_TABLE_FUNCTION',
    co_amdp_class_name   TYPE sxco_ao_object_name VALUE
'ZCL_XCO_AMDP_CLASS',

```

```

co_amdp_method_name      TYPE sxco_ao_component_name VALUE 'GET_VALUES'.
METHODS:
  add_database_table
    IMPORTING
      io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
  add_table_function
    IMPORTING
      io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put,
  add_class
    IMPORTING
      io_put_operation TYPE REF TO if_xco_cp_gen_d_o_put.
ENDCLASS.

CLASS zcl_xco_doc_cp_cs_gen_tbl_fnc IMPLEMENTATION.

  METHOD main.
    DATA(lo_put_operation) = xco_cp_generation=>environment->dev_system( co_transport
      )->create_put_operation( ).
    add_database_table( lo_put_operation ).
    add_table_function( lo_put_operation ).
    add_class( lo_put_operation ).
    lo_put_operation->execute( ).
    out->write( |Table function { co_table_function_name } generated successfully.| ).
  ENDMETHOD.

  METHOD add_database_table.
    DATA(lo_specification) = io_put_operation->for-tabl-for-database_table->add_object( co_database_table_name
      )->set_package( co_package
      )->create_form_specification( ).
    lo_specification->set_short_description( 'GENERATED' ).
    lo_specification->add_field( 'CLIENT'
      )->set_type( xco_cp_abap_dictionary=>built_in_type->clnt
      )->set_key_indicator(
      )->set_not_null( ).
    lo_specification->add_field( 'IDENTIFIER'
      )->set_type( xco_cp_abap_dictionary=>built_in_type->char( 30 )
      )->set_key_indicator(
      )->set_not_null( ).
    lo_specification->add_field( 'START_DATE'
      )->set_type( xco_cp_abap_dictionary=>built_in_type->dats ).
  ENDMETHOD.

  METHOD add_table_function.
    DATA(lo_form_specification) = io_put_operation->for-ddls->add_object( co_table_function_name
      )->set_package( co_package
      )->create_form_specification( ).
    DATA(lo_table_function) = lo_form_specification->set_short_description( 'GENERATED'
      )->add_table_function( ).
    lo_table_function->add_annotation( 'AccessControl.authorizationCheck'
      )->value->build( )->add_enum( 'NOT_REQUIRED' ).
    DATA(lo_parameter) = lo_table_function->add_parameter( 'p_identifier'
      )->set_data_type( xco_cp_abap_dictionary=>data_element( 'char30' ) ).
    DATA(lo_client_field) = lo_table_function->add_field( xco_cp_ddl=>field( 'client' ) ).
    lo_client_field->set_type( xco_cp_abap_dictionary=>built_in_type->clnt ).
    DATA(lo_val_identifier_field) = lo_table_function->add_field( xco_cp_ddl=>field( 'val_identifier' ) ).
    lo_val_identifier_field->set_type( xco_cp_abap_dictionary=>built_in_type->char( 30 ) ).
    lo_val_identifier_field->add_annotation( 'EndUserText.label'
      )->value->set( xco_cp_cds_annotation=>value->string( 'Identifier field' ) ) ##NO_TEXT.
    lo_table_function->set_amdp_class( co_amdp_class_name
      )->set_amdp_method( co_amdp_method_name ).
  ENDMETHOD.

  METHOD add_class.

```

```

DATA(lo_form_specification) = io_put_operation->for-clas-
>add_object( co_amdp_class_name
    )->set_package( co_package
    )->create_form_specification( ).
lo_form_specification->set_short_description( 'GENERATED' ).
lo_form_specification->definition->add_interface( 'IF_AMDP_MARKER_HDB' ).
DATA(lo_class_method) = lo_form_specification->definition->section-public-
>add_class_method( CONV #( co_amdp_method_name ) ).
lo_class_method->amdp->set_for_table_function( co_table_function_name ).
lo_form_specification->implementation->add_method( CONV
#( co_amdp_method_name )
    )->set_source( VALUE #(
        ( |RETURN| ) ##NO_TEXT
        ( |SELECT CLIENT, IDENTIFIER AS val_identifier FROM
{ co_database_table_name }| ) ##NO_TEXT
        ( |WHERE IDENTIFIER = :p_identifier;| ) ##NO_TEXT
    ) )->amdp->mark_as_function(
    )->set_database_type( xco_cp_amdp=>database_type->hdb
    )->set_database_language( xco_cp_amdp=>database_language->sqlscript
    )->set_database_options( VALUE #( ( xco_cp_amdp=>database_option-
>read_only ) )
    )->set_database_entities( VALUE #( ( co_database_table_name ) ) ).  

ENDMETHOD.  

ENDCLASS.

```

## 4.2.11.16.3.3.13 Generating Transformations

The XCO Generation APIs offer support for PUT and DELETE operations for transformations (XSLT objects).

The first code sample below illustrates how a transformation can be generated via a PUT operation and directly invoked dynamically to extract data from a provided XML string as well as present data as XML (making use of XCO Standard Library functionality to conveniently translate between strings and xstrings based on UTF-8).

### ↔ Sample Code

```

DATA(lv_transport_request) = CONV sxco_transport( '...' ).  

DATA(lv_transformation_name) = CONV  

sxco_tf_object_name( 'ZMY_TRANSFORMATION' ).  

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( lv_transport_request
    )->create_put_operation( ).  

DATA(lo_specification) = lo_put_operation->for-xslt-
>add_object( lv_transformation_name
    )->set_package( 'Z_MY_PACKAGE'
    )->create_form_specification( ).  

lo_specification->set_short_description( 'Generated transformation'
    )->set_source( VALUE #(
        ( |<tt:transform xmlns:tt="http://www.sap.com/transformation-
templates">| )
        ( |    <tt:root name="ROOT"/>| )
        ( |    <tt:template>| )
        ( |        <record>| )
        ( |            <tt:attribute name="text" value-ref="root.text"/>| )
        ( |        </record>| )
        ( |    </tt:template>| )
        ( |  </tt:transform>| )
    ) ).  

lo_put_operation->execute( ).  


```

```

" At this point a transformation with the name ZMY_TRANSFORMATION exists in
the system in active
" state.

TYPES:
BEGIN OF ts_record,
  text TYPE string,
END OF ts_record.
DATA ls_record TYPE ts_record.

" Using the generated transformation, it is now possible to dynamically apply
it to provided XML
" data.
DATA(lv_xml_string) = |<?xml version="1.0" encoding="UTF-8"?><record
text="Hello World"/>|.

" Converts 'string' to 'xstring' using 'UTF-8' code page.
DATA(lv_xml_xstring) = xco_cp=>string( lv_xml_string
) ->as_xstring( xco_cp_character=>code_page->utf_8
) ->value.

CALL TRANSFORMATION (lv_transformation_name)
  SOURCE XML lv_xml_xstring
  RESULT root = ls_record.

" At this point the TEXT component of the structure LS_RECORD will have the
value Hello World
" In the same manner, an XML string can be also be obtained from the ABAP
data object.
ls_record-text = 'Hello XCO'.

CALL TRANSFORMATION (lv_transformation_name)
  SOURCE root = ls_record
  RESULT XML lv_xml_xstring.

" Converts 'xstring' to 'string' using 'UTF-8' code page.
lv_xml_string = xco_cp=>xstring( lv_xml_xstring
) ->as_string( xco_cp_character=>code_page->utf_8
) ->value.

" Now the variable LV_XML_STRING will have the value <?xml version="1.0"
encoding="UTF-8"?><record text="Hello XCO"/>.

```

Just as with other object types, transformations can also be deleted:

#### ↔ Sample Code

```

DATA(lv_transport_request) = CONV sxco_transport( '...' ).
DATA(lv_transformation_name) = CONV
sxco_tf_object_name( 'ZMY_TRANSFORMATION' ).

DATA(lo_delete_operation) = xco_cp_generation=>environment-
>dev_system( lv_transport_request
) ->for-xsslt->create_delete_operation( ).
lo_delete_operation->add_object( lv_transformation_name ).
lo_delete_operation->execute( ).

```

## 4.2.11.16.3.3.14 Changing the Content of Existing Domains

Using PATCH operations, it's possible to change the content of an existing domain. The following changes can be specified:

- Changes to header-level properties, such as the format of a domain or its output length
- Insertions, updates, or deletions of fixed values

The code sample below illustrates how the following changes can be performed with a single PATCH operation:

- Update the short description, case-sensitive indicator, and output style
- Update an existing fixed value
- Delete an existing fixed value
- Insert a new fixed value

### ↔ Sample Code

```
DATA(lv_transport) = CONV sxco_transport( .... ).  
DATA(lv_domain_name) = CONV sxco_ad_object_name( 'MY_DOMAIN' ).  
  
DATA(lo_patch_operation) = xco_cp_generation=>environment->dev_system( lv_transport )->for-domains->create_patch_operation( ).  
  
DATA(lo_change_specification) = lo_patch_operation->add_object( lv_domain_name )->create_change_specification( ).  
  
" Change header-level properties.  
lo_change_specification->for-update->set_short_description( 'My new short  
description'  
    )->set_case_sensitive( abap_true  
    )->set_output_style( xco_cp_domain=>output_style->normal ).  
  
" Update the description of fixed value "A".  
lo_change_specification->for-update->add_fixed_value( 'A'  
    )->set_description( 'Fixed Value A' ).  
  
" Delete fixed value "B".  
lo_change_specification->for-delete->add_fixed_value( 'B' ).  
  
" Insert fixed value "C".  
lo_change_specification->for-insert->add_fixed_value( 'C'  
    )->set_description( 'Fixed Value C' ).  
  
lo_patch_operation->execute( ).
```

## 4.2.11.16.3.3.15 Changing the Content of Existing Data Elements

Using PATCH operations, it's possible to change the content of an existing data element. The following changes can be specified:

- Changes to header-level properties, such as the data type or the field labels

The code sample below illustrates how the following changes can be performed with a single PATCH operation:

- Update the short description
- Update the data type
- Update the short field label

#### ↳ Sample Code

```
DATA(lv_transport) = CONV sxco_transport( '...' ).  
DATA(lv_data_element_name) = CONV sxco_ad_object_name( 'MY_DATA_ELEMENT' ) .  
  
DATA(lo_patch_operation) = xco_cp_generation=>environment->dev_system( lv_transport  
    )->for-dtel->create_patch_operation( ).  
  
DATA(lo_change_specification) = lo_patch_operation->add_object( lv_data_element_name  
    )->create_change_specification( ).  
  
" Change header-level properties.  
lo_change_specification->for-update->set_short_description( 'My new short  
description'  
    )->set_data_type( xco_cp_abap_dictionary=>built_in_type->int4  
    )->field_label-short->set_text( 'Sh. FL' )->set_length( 10 ).  
  
lo_patch_operation->execute( ).
```

## 4.2.11.16.3.3.16 Generating CDS Type Definitions

The following code sample illustrates how you can generate a CDS simple type with the XCO library by means of running a PUT operation:

#### ↳ Sample Code

```
DATA(lo_put_operation) = xco_cp_generation=>environment->dev_system( '$TRANSPORT_REQUEST$'  
    )->create_put_operation( ).  
  
DATA(lo_form_specification) = lo_put_operation->for-drty->add_object( 'ZMY_CDS_TYPE_DEFINITION'  
    )->set_package( 'ZMY_PACKAGE'  
    )->create_form_specification( ).  
lo_form_specification->set_short_description( 'Generated CDS Type Definition' ).  
  
DATA(lo_simple_type_definition) = lo_form_specification->add_simple_type_definition( ).  
lo_simple_type_definition->add_annotation( 'EndUserText.label' )->value-build(  
    )->add_string( 'My CDS type definition' ).  
  
lo_simple_type_definition->set_type( xco_cp_abap_dictionary=>built_in_type->char( 10 ) ).  
  
lo_put_operation->execute( ).
```

Running this PUT operation will produce the CDS type definition ZMY\_CDS\_TYPE\_DEFINITION with the following source:

#### ↳ Sample Code

```
@EndUserText.label: 'My CDS type definition'  
define type ZMY_CDS_TYPE_DEFINITION : abap.char( 10 )
```

In general, the following objects are eligible to be used as the type for a simple type definition:

- ABAP dictionary built-in type (objects of type `CL_XCO_AD_BUILT_IN_TYPE`)
- Data elements (objects of type `IF_XCO_AD_DATA_ELEMENT`)
- CDS type definitions (objects of type `IF_XCO_CDS_TYPE_DEFINITION`)

CDS type definitions can also be deleted again, as is illustrated by the following code sample:

#### ↳ Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'  
    )->for-drty->create_delete_operation( ).  
lo_delete_operation->add_object( 'ZMY_CDS_TYPE_DEFINITION' ).  
lo_delete_operation->execute( ).
```

## 4.2.11.16.3.3.17 Patching Structures and Database Tables

Using `PATCH` operations, it's possible to change the content of an existing structure or database table. The following changes can be specified:

- Choose to update the type and/or the currency and/or the quantity information of an existing database table field or structure component
- Update of the structure of an existing include of a database table or a structure

The following code sample illustrates how the changes described above can be applied to an existing database table:

#### ↳ Sample Code

```
DATA(lv_transport) = CONV sxco_transport( '...' ).  
DATA(lv_database_table_name) = CONV sxco_dbt_object_name( 'ZMY_DBT' ).  
  
DATA(lo_patch_operation) = xco_cp_generation->environment->dev_system( lv_transport  
    )->create_patch_operation( ).  
  
DATA(lo_change_specification) = lo_patch_operation->for-tabl-for-database_table->add_object( lv_database_table_name  
    )->create_change_specification( ).  
  
    " Update the type of field DATA_FIELD to abap.char( 10 )  
    lo_change_specification->for-update->add_field( 'DATA_FIELD'  
        )->set_type( xco_cp_abap_dictionary->built_in_type->char( 10 ) ).  
  
    " Update the currency/quantity reference information for field CURR_QUAN  
    lo_change_specification->for-update->add_field( 'CURR_QUAN'  
        )->currency_quantity->set_reference_table( 'ZMY_REF_TABLE'  
        )->set_reference_field( 'ZMY_REF_FIELD' ).
```

```

" Update the structure that is used for the first include of the
" database table.
lo_change_specification->for-update->add_include( 1
    )->set_structure( 'ZMY_INCL_STR' ) .

lo_patch_operation->execute( ).
```

The same changes could also be applied to an existing structure by adding a structure to the PATCH operation created above:

#### ↔ Sample Code

```
lo_patch_operation->for-tabl-for-structure->add_object( 'ZMY_STRUCTURE' ).
```

Note that the PATCH operation obtained via

`XCO_CP_GENERATION=>ENVIRONMENT->DEV_SYSTEM( LV_TRANSPORT )->CREATE_PATCH_OPERATION( )` is a mass-enabled operation which can contain several objects at once. Upon execution, first, the specified changes will be applied to all objects that have been added to the PATCH operation, and afterwards, all objects will be activated together in a single mass activation. Currently, the following object types are enabled for mass PATCH operations:

- DOMA (Domains)
- DTEL (Data Elements)
- TABL (Database Tables and Structures)

## 4.2.11.16.3.3.18 Generating CDS External Entities

The following code sample shows how you can generate a CDS external entity with the XCO library by running a PUT operation:

#### ↔ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( 'ZMY_CDS_EXT_ENTITY'
    )->set_package( 'ZMY_PACKAGE'
    )->create_form_specification( ).

DATA(lo_external_entity) = lo_form_specification-
>set_short_description( 'Generated CDS External Entity'
    )->add_external_entity( ).

lo_external_entity->set_with_federated_data(
    )->set_dynamic_access(
    )->set_on_unlinked_fail( ).

lo_external_entity->add_annotation( 'EndUserText.label' ##NO_TEXT
    )->value->build( )->add_string( 'My external entity' ).

lo_external_entity->data_source->set_remote_object_name( 'My remote object
name' ).
```

```

lo_external_entity->add_field( xco_cp_ddl=>field( 'key_field' )
) ->set_key(
) ->set_type( xco_cp_abap_dictionary=>built_in_type->int1 ).

lo_external_entity->add_field( xco_cp_ddl=>field( 'field' )
) ->set_type( xco_cp_abap_dictionary=>built_in_type->char( 10 )
) ->set_external_name( 'My field external name' ).

lo_put_operation->execute( ).
```

Running this `PUT` operation will produce the CDS external entity `ZMY_CDS_EXT_ENTITY` with the following source:

#### ↳ Sample Code

```

@EndUserText.label: 'My external entity'
define external entity ZMY_CDS_EXT_ENTITY external name "My remote object
name"
{
    key key_field : abap.int1;
    field : abap.char(10) external name "My field external name";
}
WITH FEDERATED DATA
PROVIDED AT RUNTIME
ON UNLINKED FAIL
```

To connect to an external system, you will also need a logical external schema. It can be generated like this:

#### ↳ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( '$TRANSPORT_REQUEST$'
)->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-desd-
>add_object( 'ZMY_LOGICAL_EXT_SCHEMA'
)->set_package( 'ZMY_PACKAGE'
)->create_form_specification( ).

lo_form_specification->set_short_description( 'Generated Logical External
Schema'
)->set_default_remote_schema_name( 'My Schema Name' ).

lo_put_operation->execute( ).
```

## Related Information

[CDS External Entities](#)

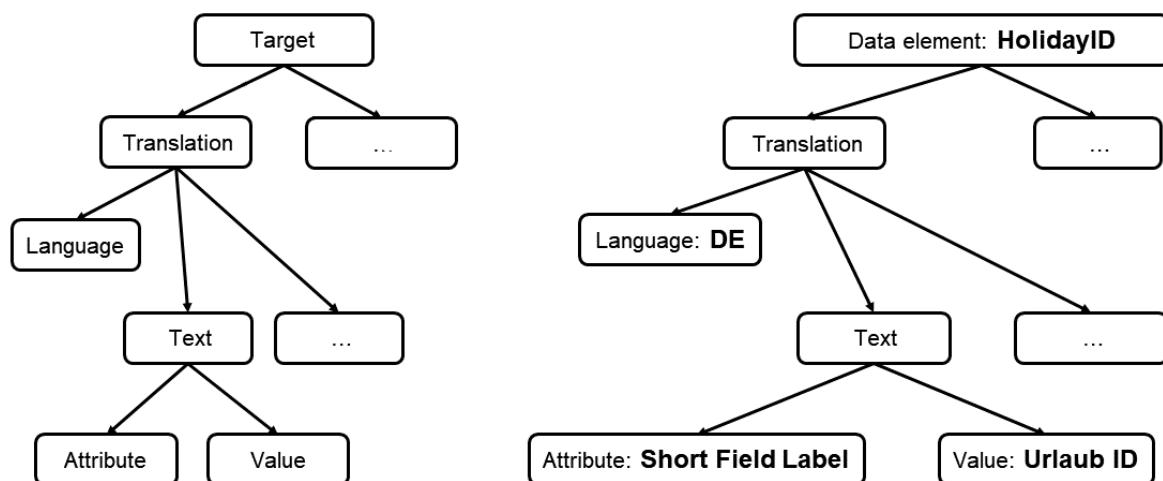
## 4.2.11.16.3.4 I18N APIs

The XCO I18N APIs allow the programmatic maintenance of translations for language-dependent texts. The I18N module consists of high-level and strongly typed APIs and provides support for the following targets:

- Domains
- Data elements
- Data definitions
- Metadata extensions
- Message classes
- Application log objects
- Business configuration objects
- IAM business catalogs
- Text pools
- Text tables

## Design of the XCO I18N APIs

The goal of the XCO I18N architecture is to provide a homogenous way to programmatically maintain and access translations for different kinds of language-dependent texts with a public API that is independent of the underlying concrete object type. The overall structure is illustrated in the following diagram:



The left side of the diagram shows the general abstractions while the right side shows what the general abstractions would correspond to in the case of the short field label of a given data element. The following terminology is used:

- Target: A collection of translations
- Translation: A tuple consisting of a language and a collection of texts
- Language: A language that is installed in the system and is identified by an SPRAS value
- Text: A tuple consisting of an attribute and a value

The following sections will describe and illustrate the available targets and text attributes in more detail:

## Domains

For a given domain a single fixed value (identified by its lower limit) defines a target. The only available text attribute is the description of a fixed value. Here's how the translation for language German could be maintained for the fixed value DE of domain ZCOUNTRY:

### Sample Code

```
DATA(lo_text_attribute) = xco_cp_domain->text_attribute->fixed_value_description.  
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Deutschland' ) ).  
DATA(lo_target) = xco_cp_i18n=>target->domain->fixed_value(  
    iv_domain_name = 'ZCOUNTRY'  
    iv_lower_limit = 'DE'  
).  
" Set the translation.  
lo_target->set_translation(  
    it_texts          = VALUE #( ( lo_text ) )  
    io_language       = lo_language  
    io_change_scenario = lo_transport_request  
).  
" Read the translation.  
DATA(lo_translation) = lo_target->get_translation(  
    io_language       = lo_language  
    it_text_attributes = VALUE #( ( lo_text_attribute ) )  
).  
LOOP AT lo_translation->texts INTO DATA(lo_german_text).  
    " LV_VALUE is of type STRING and contains `Deutschland`.  
    DATA(lv_value) = lo_text_attribute->if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).  
ENDLOOP.
```

## Data elements

Unlike domains, once a data element is fixed there are no more degrees of freedom for language-dependent texts as all the different field labels are directly defined on the level of the data element itself. The following example illustrates how the short field label of data element ZFIRST\_NAME can be translated to German (first name = Vorname) and how the translated value can be read back again:

### Sample Code

```
DATA(lo_text_attribute) = xco_cp_data_element->text_attribute->short_field_label.  
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Vorname' ) ).  
DATA(lo_target) = xco_cp_i18n=>target->data_element->object( 'ZFIRST_NAME' ).  
" Set the translation.  
lo_target->set_translation(  
    it_texts          = VALUE #( ( lo_text ) )  
    io_language       = lo_language
```

```

    io_change_scenario = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language      = lo_language
    it_text_attributes = VALUE #( ( lo_text_attribute ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Vorname`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

## Data definitions

For data definitions all language-dependent texts are supplied via annotations provided directly in the source of the data definition. A data definition defines the following targets:

- Entity
- A parameter identified by its name
- A field identified by its name

Depending on the annotation it may be necessary to specify an array index (e.g. UI.lineItem.label) to fully specify a text attribute. The APIs in XCO\_CP\_DATA\_DEFINITION=>TEXT\_ATTRIBUTE provide an easy overview of maintainable text attributes for the three different kinds of data definition targets.

The code sample below illustrates how the EndUserText.label annotation value can be translated to German for the field startDate of the CDS view entity ZVACATION\_REQUEST (start date = Startdatum):

### ↔ Sample Code

```

DATA(lo_text_attribute) = xco_cp_data_definition=>text_attribute->field-
>endusertext_label.
DATA(lo_language) = xco_cp=>language( 'D' ).
DATA(lo_text) = lo_text_attribute-
>create_text( xco_cp=>string( 'Startdatum' ) ).
DATA(iv_entity_name) = xco_cp_i18n=>target->data_definition->field(
    iv_field_name = 'ZVACATION_REQUEST'
    iv_field_name = 'startDate'
).
" Set the translation.
lo_target->set_translation(
    it_texts          = VALUE #( ( lo_text ) )
    io_language       = lo_language
    io_change_scenario = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language       = lo_language
    it_text_attributes = VALUE #( ( lo_text_attribute ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Startdatum`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

## Metadata extensions

Metadata extensions are structurally equivalent to data definitions in that all language-dependent texts are supplied via annotations provided directly in the source of the metadata extension and the available targets are

- Entity
- A parameter identified by its name
- A field identified by its name

The maintainable text attributes are available via the XCO\_CP\_METADATA\_EXTENSION=>TEXT\_ATTRIBUTE API.

The code sample below illustrates how the UI.lineitem.label annotation value can be translated to German for the field endDate of the metadata extension ZVACATION\_REQUEST\_EXT (end date = Enddatum):

### ↔ Sample Code

```
DATA(lo_text_attribute) = xco_cp_metadata_extension=>text_attribute->field->ui_lineitem_label( 1 ).  
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Enddatum' ) ).  
DATA(lo_target) = xco_cp_i18n=>target->metadata_extension->field(  
    iv_metadata_extension_name = 'ZVACATION_REQUEST_EXT'  
    iv_field_name             = 'endDate'  
).  
" Set the translation.  
lo_target->set_translation(  
    it_texts           = VALUE #( ( lo_text ) )  
    io_language        = lo_language  
    io_change_scenario = lo_transport_request  
).  
" Read the translation.  
DATA(lo_translation) = lo_target->get_translation(  
    io_language        = lo_language  
    it_text_attributes = VALUE #( ( lo_text_attribute ) )  
).  
LOOP AT lo_translation->texts INTO DATA(lo_german_text).  
    " LV_VALUE is of type STRING and contains `Enddatum`.  
    DATA(lv_value) = lo_text_attribute->if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).  
ENDLOOP.
```

## Message class

Messages classes follow a structure similar to that of domains: A single message of a message class (identified by its number) defines a target and the only available text attribute is the short text of a message.

In the code sample below message 005 of message class ZMESSAGES ("Record created." = "Eintrag angelegt.") is translated to German:

### ↔ Sample Code

```
DATA(lo_text_attribute) = xco_cp_message_class=>text_attribute->message_short_text.  
DATA(lo_language) = xco_cp=>language( 'D' ).
```

```

DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Eintrag
angelegt.' ) ).
DATA(lo_target) = xco_cp_i18n=>target->message_class->message(
    iv_message_class_name = 'ZMESSAGES'
    iv_message_number      = '005'
).
" Set the translation.
lo_target->set_translation(
    it_texts              = VALUE #( ( lo_text ) )
    io_language            = lo_language
    io_change_scenario     = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language            = lo_language
    it_text_attributes     = VALUE #( ( lo_text_attribute ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Eintrag angelegt.`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

## Application log objects

For a given application log object both the object itself as well as each of its subobjects act as targets. The only available text attribute is the short description. The following code sample illustrates how the German translation for the short description of the application log object Z\_PROGRESS\_RECORDING can be set and read:

### ↔ Sample Code

```

DATA(lo_text_attribute) = xco_cp_application_log_object->text_attribute-
>object->short_description.

DATA(lo_language) = xco_cp=>language( 'D' ).
DATA(lo_text) = lo_text_attribute-
>create_text( xco_cp=>string( 'Fortschrittsaufzeichnung' ) ).

DATA(lo_target) = xco_cp_i18n=>target->application_log_object-
>object( 'Z_PROGRESS_RECORDING' ).

" Set the translation.
lo_target->set_translation(
    it_texts              = VALUE #( ( lo_text ) )
    io_language            = lo_language
    io_change_scenario     = lo_transport_request
).

" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language            = lo_language
    it_text_attributes     = VALUE #( ( lo_text_attribute ) )
).

LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Fortschrittsaufzeichnung`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

Setting and getting translations for subobjects works in completely analogous fashion by first obtaining the subobject target and then using text attribute XCO\_CP\_APPLICATION\_LOG\_OBJECT=>TEXT\_ATTRIBUTE->SUBOBJECT->SHORT\_DESCRIPTION.

## Business configuration objects

Translations can also be maintained for business configuration objects which are available for maintenance via the "Maintain Business Configurations" Fiori app. Each business configuration object has a name and description which is visible in the Maintain Business Configurations app. The code sample below illustrates how the description of a business configuration object can be translated (and read out).

### ↔ Sample Code

```
DATA(lo_text_attribute) = xco_cp_business_cnfgrtn_object=>text_attribute->description.  
  
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Urlaubskalender' ) ).  
  
DATA(lo_target) = xco_cp_i18n=>target->business_configuration_object-object( 'Z_HOLIDAY_CALENDAR' ).  
  
" Set the translation.  
lo_target->set_translation(  
    it_texts           = VALUE #( ( lo_text ) )  
    io_language        = lo_language  
    io_change_scenario = lo_transport_request  
).  
  
" Read the translation.  
DATA(lo_translation) = lo_target->get_translation(  
    io_language        = lo_language  
    it_text_attributes = VALUE #( ( lo_text_attribute ) )  
).  
  
LOOP AT lo_translation->texts INTO DATA(lo_german_text).  
    " LV_VALUE is of type STRING and contains `Urlaubskalender`.  
    DATA(lv_value) = lo_text_attribute->if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).  
ENDLOOP.
```

## IAM business catalogs

Translations can also be maintained for IAM business catalogs. The only text attribute of an IAM business catalog is its description. The code sample below illustrates how the description of an IAM business catalog can be translated (and read out).

### ↔ Sample Code

```
DATA(lo_text_attribute) = xco_cp_iam_business_catalog=>text_attribute->description.  
DATA(lo_language) = xco_cp=>language( 'D' ).
```

```

DATA(lo_text) = lo_text_attribute-
>create_text( xco_cp=>string( 'Buchhaltung' ) ).
DATA(lo_target) = xco_cp_iam=>business_catalog->for( 'Z_ACCOUNTING' )-
>i18n_target->object( ).
" Set the translation.
lo_target->set_translation(
    it_texts          = VALUE #( ( lo_text ) )
    io_language       = lo_language
    io_change_scenario = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language       = lo_language
    it_text_attributes = VALUE #( ( lo_text_attribute ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Buchhaltung`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

## Text pools

Both classes and function groups allow the definition of text symbols which are stored in the text pool associated with the respective class or function group. Maintaining the translations for such text symbols can be accomplished like:

### ↔ Sample Code

```

DATA(lo_text_attribute) = xco_cp_text_pool=>text_attribute->text_element_text.
DATA(lo_language) = xco_cp=>language( 'D' ).
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Auftrag
abgeschickt.' ) ).
DATA(lo_target) = xco_cp_i18n=>target->text_pool->class_text_symbol(
    iv_class_name      = 'ZCL_MY_CLASS'
    iv_text_symbol_id = '001'
).
" Set the translation.
lo_target->set_translation(
    it_texts          = VALUE #( ( lo_text ) )
    io_language       = lo_language
    io_change_scenario = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language       = lo_language
    it_text_attributes = VALUE #( ( lo_text_attribute ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Auftrag abgeschickt.`.
    DATA(lv_value) = lo_text_attribute-
>if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).
ENDLOOP.

```

In the above example the German translation is maintained for the text of text symbol 001 (the original text is "Request sent.") of class ZCL\_MY\_CLASS. If the text symbol belonged to a function group instead of a class the code would look like:

## ↳ Sample Code

```
DATA(lo_text_attribute) = xco_cp_text_pool->text_attribute->text_element_text.  
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = lo_text_attribute->create_text( xco_cp=>string( 'Auftrag  
abgeschickt.' ) ).  
DATA(lo_target) = xco_cp_i18n=>target->text_pool->function_group_text_symbol  
    iv_function_group_name = 'ZMY_FNCTN_GRP'  
    iv_text_symbol_id      = '001'  
).  
" Set the translation.  
lo_target->set_translation(  
    it_texts          = VALUE #( ( lo_text ) )  
    io_language       = lo_language  
    io_change_scenario = lo_transport_request  
).  
" Read the translation.  
DATA(lo_translation) = lo_target->get_translation(  
    io_language       = lo_language  
    it_text_attributes = VALUE #( ( lo_text_attribute ) )  
).  
LOOP AT lo_translation->texts INTO DATA(lo_german_text).  
    " LV_VALUE is of type STRING and contains `Auftrag abgeschickt.`.  
    DATA(lv_value) = lo_text_attribute-  
        >if_xco_i18n_text_attribute~get_string_for_text( lo_german_text->value ).  
ENDLOOP.
```

## Text tables

Any database table with delivery class C or S which has exactly one key field with the underlying ABAP Dictionary built-in type LANG is considered a text table and can act as a source of targets. Each key field of a text table is uniquely associated with one of the following three groups:

- Client key field
- Master key fields
- Language key field

The semantics are such that a full specification of all master key fields of a text table defines a target and that each non-key field which is character-like and has at least length 2 is considered a text attribute of that target.

As an example, consider the database table ZXCO\_CAL\_HOLITXT which contains the language-dependent texts for public holidays maintained in database table ZXCO\_CAL\_HLDY:

{ZXCO\_CAL\_HLDY}

{ZXCO\_CAL\_HOLITXT}

Here CLIENT is the client key field of the text table ZXCO\_CAL\_HOLITXT, HOLIDAY\_ID is the master key field and LANGUAGE is the language key field. Fixing a value for the HOLIDAY\_ID key field (e.g. "CHRISTMAS") determines a target for which the text attribute DESCRIPTION can then be read and written, as is illustrated by the code sample below:

## ↳ Sample Code

```
DATA(lo_language) = xco_cp=>language( 'D' ).  
DATA(lo_text) = xco_cp=>string( 'Weihnachten' ).
```

```

DATA(lo_target) = xco_cp_i18n->target->text_table->record(
    iv_database_table_name      = 'ZCAL_XCO_HOLITXT'
    it_master_key_fields       = VALUE #( ( name = 'HOLIDAY_ID' value =
xco_cp->data_object->for( 'CHRISTMAS' ) ) )
    iv_language_key_field_name = 'LANGUAGE'
).
" Set the translation.
lo_target->set_translation(
    it_texts                  = VALUE #( ( attribute = 'DESCRIPTION' value =
lo_text ) )
    io_language               = lo_language
    io_change_scenario        = lo_transport_request
).
" Read the translation.
DATA(lo_translation) = lo_target->get_translation(
    io_language               = lo_language
    it_text_attributes        = VALUE #( ( 'DESCRIPTION' ) )
).
LOOP AT lo_translation->texts INTO DATA(lo_german_text).
    " LV_VALUE is of type STRING and contains `Weihnachten`.
    DATA(lv_value) = lo_german_text->get_string_value( ).
ENDLOOP.

```

## 4.2.11.16.3.5 API Release Framework

Find out how to set the API state.

Setting and getting API states programmatically is supported for the following objects:

- Classes (IF\_XCO\_AO\_CLASS)
- Entities of data definitions (IF\_XCO\_CDS\_ENTITY)
- Domains (IF\_XCO\_DOMAIN)
- Data elements (IF\_XCO\_AD\_DATA\_ELEMENT)
- Function modules of function groups (IF\_XCO\_FUNCTION\_MODULE)
- Interfaces (IF\_XCO\_AO\_INTERFACE)
- Structures (IF\_XCO\_AD\_STRUCTURE)
- Database tables (IF\_XCO\_DATABASE\_TABLE)
- Table types (IF\_XCO\_AD\_TABLE\_TYPE)
- Transformations (IF\_XCO\_TRANSFORMATION)

Each of the above interfaces defines the two methods SET\_API\_STATE and GET\_API\_STATE to programmatically set or get the API state of the represented object. For data elements, for example, this looks like:

### ↳ Sample Code

```

DATA(lo_transport) = xco_cp_cts->transport->for( '...' ).
DATA(lo_api_state) = xco_cp_ars->api_state->released( VALUE
#( ( xco_cp_ars=>visibility->sap_cloud_platform ) ) ).

DATA(lo_data_element) = xco_cp_abap_repository->object->dtel-
>for( 'ZMY_DATA_ELEMENT' ).
lo_data_element->set_api_state(
    io_change_scenario = lo_transport
    io_api_state      = lo_api_state

```

) .

### 4.2.11.16.3.6 Core Data Services

The XCO library provides several abstractions and APIs to simplify working with objects in the area of Core Data Services. The functionality ranges from reading the content of CDS entities to querying annotations.

#### Read APIs

With the CDS Read APIs it is possible to access the content of a given CDS entity in a strongly typed manner, as is illustrated by the following example:

##### ↳ Sample Code

```
DATA(lo_view_entity) = xco_cp_cds->view_entity( 'MY_VIEW_ENTITY' ) .
DATA(lo_view_entity_content) = lo_view_entity->content( ) .
DATA(ls_view_entity_content) = lo_view_entity_content->get( ) .
DATA(lv_short_description) = ls_view_entity_content-short_description .
DATA(ls_data_source) = ls_view_entity_content-data_source .
DATA(lv_root_indicator) = ls_view_entity_content-root_indicator .
DATA(lt_name_list) = ls_view_entity_content-name_list .
DATA(lo_where) = ls_view_entity_content-where .
DATA(lt_group_by) = ls_view_entity_content-group_by .
" Attributes can be accessed directly via corresponding get methods, such as:
lt_name_list = lo_view_entity_content->get_name_list( ) .
LOOP AT lo_view_entity->parameters->all->get( ) INTO DATA(lo_parameter) .
  DATA(ls_parameter) = lo_parameter->content( )->get( ) .
ENDLOOP .
LOOP AT lo_view_entity->associations->all->get( ) INTO DATA(lo_association) .
  DATA(ls_association) = lo_association->content( )->get( ) .
ENDLOOP .
LOOP AT lo_view_entity->compositions->all->get( ) INTO DATA(lo_composition) .
  DATA(ls_composition) = lo_composition->content( )->get( ) .
ENDLOOP .
LOOP AT lo_view_entity->fields->all->get( ) INTO DATA(lo_field) .
  DATA(ls_field) = lo_field->content( )->get( ) .
ENDLOOP .
" Fields/parameters/associations/compositions can be addressed directly, such
as:
lo_field = lo_view_entity->field( 'FIELD_1' ) .
DATA(lv_does_field_exist) = lo_field->exists( ) .
IF lv_does_field_exist EQ abap_true .
  ls_field = lo_field->content( )->get( ) .
ENDIF .
```

#### Data Definition Language

Using the Data Definition Language (DDL) module of the XCO Library, different kinds of DDL expressions can be built and successively used in conjunction with the XCO Generation APIs when providing specifications for

DDLS objects. Complex expressions like case, cast, or conditional expressions as well as literals, field, and data source expressions can be easily specified using the XCO\_CP\_DDL API:

#### ↳ Sample Code

```
" Case expression
DATA(lo_case_expression_builder) = xco_cp_ddl->expression->case->builder( ).
lo_case_expression_builder->set_operand( xco_cp_ddl->field( 'NAME' )
)->add_when(
    io_operand = xco_cp_ddl->expression->for( 'JOHN' )
    io_result = xco_cp_ddl->literal->numeric( 1 )
)->add_when(
    io_operand = xco_cp_ddl->expression->for( 'MARK' )
    io_result = xco_cp_ddl->literal->numeric( 2 )
)->add_when(
    io_operand = xco_cp_ddl->expression->for( 'JULIA' )
    io_result = xco_cp_ddl->literal->numeric( 3 )
)->set_else( xco_cp_ddl->literal->numeric( 4 ) ).

" Conditional expression
DATA(lo_condition_expression) = xco_cp_ddl->expression->for( 'CITY'
)->ne( xco_cp_ddl->literal->character( 'BERLIN' )
)->or( xco_cp_ddl->expression->for( 'CITY'
)->is_not_initial( ) ).

" Data source expression
DATA(lo_data_source_expression) = xco_cp_ddl->data_source-
>entity( 'XCO_TEST_ENTITY'
)->inner_join(
    io_data_source = xco_cp_ddl->data_source-
>database_table( 'XCO_TEST_TABLE' )
    io_condition = xco_cp_ddl->field( 'FIELD' )->of_projection(
        )->eq( xco_cp_ddl->field( 'FIELD' )->of( 'XCO_TEST_TABLE' )
)
).
).
```

## Annotations Query APIs

The XCO CDS module may be used to conveniently retrieve the value of annotations that have been provided for CDS entities as well as their fields or parameters. As with the XCO JSON module it is possible to write the annotation value to an explicitly defined ABAP structure to enable further processing of the value:

#### ↳ Sample Code

```
TYPES:
BEGIN OF ts_title,
    type TYPE string,
    label TYPE string,
    value TYPE string,
END OF ts_title,
BEGIN OF ts_header_info,
    typename      TYPE string,
    typenameplural TYPE string,
    title         TYPE ts_title,
END OF ts_header_info,
BEGIN OF ts_ui,
    headerinfo TYPE ts_header_info,
END OF ts_ui.
DATA ls_ui TYPE ts_ui.
" Direct annotations.
DATA(lo_view_entity) = xco_cp_cds->view_entity( 'MY_VIEW_ENTITY' ).
```

```
xco_cp_cds=>annotations->direct->of( lo_view_entity  
 )->pick( 'UI'  
 )->get_value(  
 )->write_to( REF #( ls_ui ) ).
```

In the example above, the value of the “UI” annotation defined directly on the provided CDS view entity is read and written to the specifically defined ABAP structure. Parts of the annotation value not specified in the ABAP structure will be ignored.

Annotations for CDS entities are aggregated at runtime from different sources. With the XCO annotation Query APIs it is easy to specify which source should be considered when annotations are retrieved, e.g. it is possible to consider only annotations defined in metadata extensions:

#### ↔ Sample Code

```
DATA(lo_view_entity_field) = xco_cp_cds=>view_entity( 'MY_VIEW_ENTITY'  
 )->field( 'MY_FIELD' ).  
DATA(lv_ui_annotation_contained) = xco_cp_cds=>annotations-  
>metadata_extension->of( lo_view_entity_field  
 )->contain( 'UI' ).
```

The following annotation sources are available via XCO\_C\_P\_CDS=>ANNOTATIONS:

- Aggregated: The aggregation of all the different sources according to the preference rules reflecting the value that will be present at runtime
- Derived: Annotations that are derived from data elements
- Direct: Annotations that are defined directly in the source for the given CDS entity
- Inherited: Annotations that are inherited from a CDS entity that the given CDS entity is based on
- Metadata extension: Annotations that are defined in a metadata extension that extends the given CDS entity

## 4.2.11.16.3.7 Correction and Transport System

The XCO CTS module provides APIs which support the creation and release of Workbench and Customizing transports and integrates seamlessly into the XCO ABAP Repository module to support locating objects contained on transports.

Creation and releasing transport requests programmatically is especially useful in combination with the XCO Generation APIs to build completely self-contained generation programs.

### Creating and releasing Workbench transport requests

A prerequisite for creating a new Workbench transport is a valid transport target which is usually derived from the structural package for which objects are supposed to be created, changed or deleted in a corresponding development subpackage.

Using the XCO ABAP Repository APIs the transport layer and transport target associated with a given structure package can be determined easily:

#### ↳ Sample Code

```
DATA(ls_package) = xco_cp_abap_repository->package->for( 'ZMY_STRUCTURE_PACKAGE' )->read( ).  
DATA(lv_transport_layer) = ls_package->property-transport_layer->value.  
DATA(lv_transport_target) = ls_package->property-transport_layer->get_transport_target( )->value.
```

Once the correct transport target has been determined a new transport request can be created like this:

#### ↳ Sample Code

```
DATA(lo_transport_request) = xco_cp_cts->transports->workbench( lv_transport_target )->create_request( 'My generated Workbench transport request' ).
```

This statement will create a new workbench transport request with one unclassified task in the name of the active user. Once all required changes have been recorded on the tasks of the transport request releasing the transport request along with all open tasks is performed as follows:

#### ↳ Sample Code

```
DATA(lt_transport_tasks) = lo_transport_request->get_tasks( ).  
LOOP AT lt_transport_tasks INTO DATA(lo_transport_task).  
  CHECK lo_transport_task->get_status( ) EQ xco_cp_transport->status->modifiable.  
  lo_transport_task->release( ).  
ENDLOOP.  
lo_transport_request->release( ).
```

When releasing a transport, it's also possible to specify dedicated release options that should be considered when the transport is released. Transport release options are accessible via `XCO_CP_TRANSPORT=>RELEASE_OPTION` and as of now, the following transport release options are offered:

- `IGNORE_OBJECTS_CHECK`: If non-critical issues are detected with objects contained on the transport (such as issues due to ATC violations), this option allows you to ignore these issues and release the transport anyways. Setting this option has an effect equivalent to explicitly confirming the release of the transport when an interactive tool, such as the ADT Transport Organizer, is used to release the transport.

Releasing a transport request with the ignore objects check option can be done like this:

#### ↳ Sample Code

```
DATA(lo_ignore_objects_check_option) = xco_cp_transport->release_option->ignore_objects_check( ).  
lo_transport_request->release( VALUE #( ( lo_ignore_objects_check_option ) ) ).
```

Besides being able to easily release transport tasks and requests, it is also possible to protect and unprotect transport requests:

### ↔ Sample Code

```
" Protect a transport request.  
lo_transport_request->protect( ).  
" Unprotect a transport request.  
lo_transport_request->unprotect( ).
```

## Creating Customizing transport requests

A Customizing transport request for a given transport target can be created as follows:

### ↔ Sample Code

```
DATA(lo_transport_request) = xco_cp_cts->transports->customizing( '$TRANSPORT_TARGET$'  
    )->create_request( 'My generated Customizing transport request' ).
```

Once created, a Customizing transport request can be further used in the context of Business Configuration, e.g. the *Export Customizing Transports* Fiori app.

## Querying and reading transports

Within the XCO Library, a transport (IF\_XCO\_CP\_TRANSPORT) represents either a transport request or a task of a transport request. One of the characteristic properties of a transport is that it always has an associated request which is either the transport itself in case it already represents a transport request or otherwise the transport request of the task represented by the transport.

In a style similar to the ABAP Repository Query APIs it is possible to efficiently obtain a list of transports matching user-defined filters. Filters are obtained via the XCO\_CP\_TRANSPORT=>FILTER API and currently cover the following aspects of transports:

- Kind (Request or Task)
- Owner
- Request
- Target of the request
- Type of the request
- Status
- Type
- Attributes

Specific to the CTS transport Query APIs is the concept of a resolution that is applied to a transport query. A resolution defines a mapping of the matched transports to the final list of transports. Currently, the following resolutions are offered:

- Request: Map each matched transport to its associated request
- Identity: Map each matched transport to itself

As a first example, obtaining a list of transport requests for a given target on which the current user has at least one modifiable task can be accomplished like:

#### ↳ Sample Code

```
DATA(lo_current_user) = xco_cp=>sy->user( ).  
DATA(lo_kind_filter) = xco_cp_transport=>filter->kind( xco_cp_transport=>kind->task ).  
DATA(lo_owner_filter) = xco_cp_transport=>filter->owner( xco_cp_abap_sql=>constraint->equal( lo_current_user->name ) ).  
DATA(lo_request_target_filter) = xco_cp_transport=>filter->request_target( xco_cp_abap_sql=>constraint->equal( 'ZX11' ) ).  
DATA(lo_status_filter) = xco_cp_transport=>filter->status( xco_cp_transport=>status->modifiable ).  
DATA(lt_transports) = xco_cp_cts=>transports->where( VALUE #(  
    ( lo_kind_filter )  
    ( lo_owner_filter )  
    ( lo_request_target_filter )  
    ( lo_status_filter )  
) )->resolve( xco_cp_transport=>resolution->request ).  
LOOP AT lt_transports INTO DATA(lo_transport).  
    DATA(lo_transport_request) = lo_transport->get_request( ).  
    " LO_TRANSPORT is of type IF_XCO_CP_TRANSPORT. Based on the applied  
    resolution  
    " it is guaranteed that every LO_TRANSPORT object already represents a  
    request.  
ENDLOOP.
```

As a second example, obtaining a list of all transport requests which are both released and have the attribute SAP\_CUS\_TRANSPORT\_CATEGORY with the value DEFAULT\_CUS can be accomplished like:

#### ↳ Sample Code

```
DATA(lo_attribute_filter) = xco_cp_transport=>filter->attribute(  
    io_name_constraint = xco_cp_abap_sql=>constraint->equal( 'SAP_CUS_TRANSPORT_CATEGORY' )  
    io_value_constraint = xco_cp_abap_sql=>constraint->equal( 'DEFAULT_CUS' )  
).  
DATA(lo_status_filter) = xco_cp_transport=>filter->status( xco_cp_transport=>status->released ).  
DATA(lt_transport_requests) = xco_cp_cts=>transports->where( VALUE #(  
    ( lo_attribute_filter )  
    ( lo_status_filter )  
) )->resolve( xco_cp_transport=>resolution->identity ).  
LOOP AT lt_transport_requests INTO DATA(lo_transport_request).  
    " LV_TRANSPORT_REQUEST will be a transport request which is both in status  
    "Released"  
    " and has attribute SAP_CUS_TRANSPORT_CATEGORY set to value DEFAULT_CUS.  
    DATA(lv_transport_request) = lo_transport_request->value.  
ENDLOOP.
```

## Integration with the ABAP Repository APIs

The XCO CTS module integrates directly into the XCO ABAP Repository module which allows to take advantage of the powerful filtering mechanisms provided by the XCO ABAP Repository module. To this extent, the type IF\_XCO\_CP\_TRANSPORT implements the IF\_XCO\_AR\_OBJECT\_SOURCE interface.

With this integration, locating all domains on a given transport is as easy as

#### ↔ Sample Code

```
DATA(lo_transport) = xco_cp_cts->transport->for( '...' ).  
DATA(lt_domains) = xco_cp_abap_repository->objects->doma->all-  
in( lo_transport )->get( ).
```

All filters that can be applied when objects are being queried in a package or the whole ABAP Repository can also be applied when a transport is used as the object source.

Conversely, it is also possible to check if a given ABAP Repository object is currently locked on a transport request:

#### ↔ Sample Code

```
DATA(lo_domain) = xco_cp_abap_repository->object->doma->for( 'ZMY_DOMAIN' ).  
IF lo_domain->if_xco_cts_changeable~get_object( )->is_locked( ) EQ abap_true.  
    DATA(lv_transport) = lo_domain->if_xco_cts_changeable~get_object( )  
        ->get_lock( ).  
        ->get_transport( ).  
    " LO_TRANSPORT_REQUEST will be the transport request that ZMY_DOMAIN is  
    " currently locked on (if it is locked at all).  
    DATA(lo_transport_request) = xco_cp_cts->transport->for( lv_transport )  
        ->get_request( ).  
ENDIF.
```

## 4.2.11.16.3.7.1 Transport Read APIs

### Reading properties of a transport

Given a transport request (represented as an object of type `IF_XCO_CP_TR_REQUEST`), its properties can be read out as follows:

#### ↔ Sample Code

```
DATA(ls_properties) = lo_transport_request->properties( )->get( ).  
" LS_PROPERTIES is a structure providing the short description, owner, target,  
" status and last changed moment of the transport request.
```

The properties of a transport task (represented as an object of type `IF_XCO_CP_TR_TASK`) can similarly be read out like this:

#### ↔ Sample Code

```
DATA(ls_properties) = lo_transport_task->properties( )->get( ).  
" LS_PROPERTIES is a structure providing the short description, owner,  
" status and last changed moment of the transport task.
```

Properties of both transport requests and transport tasks can also be retrieved individually by using one of the specialized GET\_ methods, e.g. GET\_SHORT\_DESCRIPTION, instead of the general GET method which retrieves all available properties.

## Reading attributes of a transport request

Given a transport request (represented as an object of the type IF\_XCO\_CP\_TR\_REQUEST), its attributes can be read out as follows:

### ↔ Sample Code

```
DATA(lt_attributes) = lo_transport_request->attributes->all->get( ).  
LOOP AT lt_attributes INTO DATA(lo_attribute).  
  " The name of the attribute (as a CHAR of length 30).  
  DATA(lv_name) = lo_attribute->get_attribute( )->name.  
  " The value of the attribute (as a CHAR of length 32).  
  DATA(lv_value) = lo_attribute->get_value( ).  
ENDLOOP.
```

## Reading entries of a transport

A transport can contain two kinds of entries:

- Object entries: An object entry of a transport is identified by its values for Program ID (R3TR, LIMU or LANG), its object type and object name
- Key entries: A key entry of a transport identifies one or several rows in a database table that are to be transported once the transport is released. A key entry always has a corresponding object entry on the same transport

Both object and key entries of a transport can easily be retrieved via the XCO transport module. The starting point for retrieving entries of a transport is XCO\_CP\_TRANSPORT->ENTRIES.

## Reading object entries

The following code sample shows how to read all object entries contained on a given transport:

### ↔ Sample Code

```
" Get all object entries on a transport (given as an object LO_TRANSPORT of  
type IF_XCO_CP_TRANSPORT).  
DATA(lt_object_entries) = xco_cp_transport=>entries->object->all-  
>on( lo_transport )->get( ).  
  
LOOP AT lt_object_entries INTO DATA(lo_object_entry).  
  " The Program ID of the object entry represented as an XCO enumeration  
  constant  
  " of type CL_XCO_CTS_PROGRAM_ID. The primitive value for the program ID can  
  be  
  " obtained via LO_PROGRAM_ID->VALUE.  
  DATA(lo_program_id) = lo_object_entry->object->program_id.
```

```

    " The object type of the object entry (e.g. DOMA for a domain of the ABAP
Dictionary).
    DATA(lv_object_type) = lo_object_entry->object->type.

    " The object name of the object entry (e.g. the name of the domain if the
object entry
    " refers to a domain of the ABAP Dictionary).
    DATA(lv_object_name) = lo_object_entry->object->name.
ENDLOOP.
```

In addition to retrieving all objects it is furthermore possible to provide one or more entry filters to restrict the object entries to those matching the provided entry filters:

#### ↳ Sample Code

```

" The program ID filter will only match object entries where the program ID
is LIMU.
DATA(lo_program_id_filter) = xco_cp_transport->entry_filter->object-
>program_id(
    xco_cp_cts=>program_id->limu
).

" The object type filter will only match object entries where the object type
is FUNC.
DATA(lo_object_type_filter) = xco_cp_transport->entry_filter->object-
>object_type(
    xco_cp_abap_sql=>constraint->equal( 'FUNC' )
).

" The object name filter will only match object entries where the object name
starts with Z.
DATA(lo_object_name_filter) = xco_cp_transport->entry_filter->object-
>object_name(
    xco_cp_abap_sql=>constraint->contains_pattern( 'Z%' )
).

" LT_OBJECT_ENTRIES will contain all LIMU FUNC object entries on LO_TRANSPORT
where the object name starts with Z
" (where LO_TRANSPORT is an object of type IF_XCO_CP_TRANSPORT).
DATA(lt_object_entries) = xco_cp_transport->entries->object->where( VALUE #(
    ( lo_program_id_filter )
    ( lo_object_type_filter )
    ( lo_object_name_filter )
) )->on( lo_transport )->get( ).
```

#### Reading key entries

Reading key entries on a transport works in a similar manner as when object entries are being read. Obtaining all key entries on a given transport can be accomplished like this:

#### ↳ Sample Code

```

" Get all key entries on a transport (given as an object LO_TRANSPORT of type
IF_XCO_CP_TRANSPORT).
DATA(lt_key_entries) = xco_cp_transport->entries->key->all-
>on( lo_transport )->get( ).

LOOP AT lt_key_entries INTO DATA(lo_key_entry).
    " Get the object entry to which this key entry belongs.
    DATA(lo_object_entry) = lo_key_entry->get_object_entry( ).

    " Get the table key for the entry which can be used to subsequently retrieve
    " the column values used to select the database table entries that shall be
```

```

" transported.
DATA(lo_table_key) = lo_key_entry->get_table_key( ).

DATA(lt_table_key_components) = lo_table_key->get_components( ).

LOOP AT lt_table_key_components INTO DATA(lo_table_key_component).
  " The name of the field in the underlying database table.
  DATA(lv_field_name) = lo_table_key_component->field_name.

  " The value which is used for this field. The WRITE_TO method of
  " LO_VALUE can be used to write the value for the field to an actual
  " data object.
  DATA(lo_value) = lo_table_key_component->get_value( ).

  " The type of the value depends on the type of the field in the database
  table.
  " In the case where it is of type C of length 10, the actual value can be
  " obtained like this:
  DATA lv_value TYPE C LENGTH 10.

  lo_value->write_to( REF #( lv_value ) ).

ENDLOOP.
ENDLOOP.

```

An R3TR TABU object entry always has exactly one corresponding key entry. Thanks to a direct integration of R3TR TABUs with the entry query APIs, all key entries coming from R3TR TABU object entries can be conveniently retrieved as follows:

#### ↔ Sample Code

```

" Get all R3TR TABU entries on a transport (given as an object LO_TRANSPORT
of type IF_XCO_CP_TRANSPORT).
DATA(lt_r3tr_tabu_entries) = xco_cp_transport->entries->object->r3tr->tabu-
>all->on( lo_transport )->get( ).

LOOP AT lt_r3tr_tabu_entries INTO DATA(lo_r3tr_tabu_entry).
  " The key entry corresponding to this R3TR TABU object entry.
  DATA(lo_key_entry) = lo_r3tr_tabu_entry->get_key_entry( ).

ENDLOOP.

```

## 4.2.11.16.3.8 Object Type References

### 4.2.11.16.3.8.1 CDS Type Definitions

Find out how to create CDS type definitions.

CDS type definitions, meaning objects of type DRTY, are supported by the following API families within the XCO library:

- Query APIs
- Read APIs

- Generation APIs

## Query APIs

Obtaining an object handle for a CDS type definition with a given name can be accomplished via

### ↳ Sample Code

```
DATA(lo_cds_type_definition) = xco_cp_abap_repository->object->drty-
>for( '$CDS_TYPE_DEFINITION_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the CDS type definition in the ABAP repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible CDS type definitions in the ABAP repository can be done via

### ↳ Sample Code

```
DATA(lt_cds_type_definitions) = xco_cp_abap_repository->objects->drty->all-
>in( xco_cp_abap=>repository )->get( ).

LOOP AT lt_cds_type_definitions ASSIGNING FIELD-
SYMBOL(<fs_cds_type_definition>).
  " Process the found CDS type definition.
ENDLOOP.
```

Instead of searching for CDS type definitions in the entire ABAP repository, it's also possible to only search for CDS type definitions in a dedicated subset of the ABAP repository, such as in a single package:

### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).

DATA(lt_cds_type_definitions) = xco_cp_abap_repository->objects->drty->all-
>in( lo_package )->get( ).

LOOP AT lt_cds_type_definitions ASSIGNING FIELD-
SYMBOL(<fs_cds_type_definition>).
  " Process the found CDS type definition.
ENDLOOP.
```

Furthermore, you can refine a search for CDS type definitions by using filters to specify exactly which CDS type definitions should be matched, for example:

### ↳ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(
  xco_cp_abap_sql=>constraint->contains_pattern( '/ABC/%' )
).

" LT_CDS_TYPE_DEFINITIONS will be a list of all visible CDS type definitions
" in the ABAP Repository whose name starts with /ABC/.
DATA(lt_cds_type_definitions) = xco_cp_abap_repository->objects->drty-
>where( VALUE #(
  lo_object_name_filter )
```

```

) )->in( xco_cp_abap=>repository )->get( ).

LOOP AT lt_cds_type_definitions ASSIGNING FIELD-
SYMBOL(<fs_cds_type_definition>).
    " Process the found CDS type definition.
ENDLOOP.

```

## Read APIs

Reading out the content and checking the existence of a CDS type definition is possible via its object handle `IF_XCO_CDS_TYPE_DEFINITION`. The Read APIs for CDS type definitions are based on the origin infrastructure with the following origins being offered currently for CDS type definitions:

- **DEFAULT**: Can be obtained via `XCO_CP_CDS_TYPE_DEFINITION=>ORIGIN=DEFAULT` and will read a given CDS type definition exactly once per ABAP session from the local system, with all subsequent reads of the CDS type definition being fulfilled from a dedicated buffer. This origin is used by default, meaning if no origin is explicitly specified when reading the content or checking the existence of a CDS type definition.
- **LOCAL**: Can be obtained via `XCO_CP_CDS_TYPE_DEFINITION=>ORIGIN=LOCAL` and will read a given CDS type definition from the local system. By default, every read of the content as well as existence checks of a CDS type definition will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_DRTY_CACHEABLE_ORIGIN`.

Next, when reading a CDS type definition or checking its existence, it can explicitly be specified which version of the CDS type definition should be considered. The following versions are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_CDS_TYPE_DEFINITION=>VERSION=ACTIVE`. The active version is used by default, meaning if no version is explicitly specified when reading the content of a CDS type definition or checking its existence.
- **INACTIVE**: Can be obtained via `XCO_CP_CDS_TYPE_DEFINITION=>VERSION=INACTIVE`.

Following the general style of the XCO Read APIs, each part of a CDS type definition can be read out separately:

### Header

Reading out the header of a CDS type definition can be done like this:

#### ↔ Sample Code

```

DATA(lo_cds_type_definition) = xco_cp_abap_repository=>object->drtyst-
>for( '$CDS_TYPE_DEFINITION_NAME$' ).

" Read out the complete header of the active version of the CDS type
" definition using the default origin:
DATA(ls_header) = lo_cds_type_definition->content( )->get( ).

" Read out only the short description of the inactive version of the CDS
" type definition using an explicitly obtained local origin (without
" caching).
DATA(lo_local_origin) = xco_cp_cds_type_definition=>origin->local( ).

DATA(lv_short_description) = lo_cds_type_definition-
>content( xco_cp_cds_type_definition=>version->inactive
)->get_short_description( lo_local_origin ).

```

## Generation APIs

CDS type definitions are integrated with the XCO Generation APIs with the following operations being offered currently for CDS type definitions:

- **PUT**: Create or update a CDS type definition based on an entire content specification so that after running the operation successfully, the CDS type definition exists and its content is exactly as specified
- **DELETE**: Delete a CDS type definition (if it exists)

### PUT Operations

A form specification is used to specify the entire content of a CDS type definition, which is the basis for running a PUT operation. Upon running the PUT operation, depending on whether the CDS type definition already exists or not, it will either be newly created, or updated. This is so that after the successful completion of the PUT operation, the CDS type definition exists with the content resembling exactly what was specified in the provided form specification. Consider the following code sample which illustrates how to run a PUT operation for a CDS type definition:

#### Sample Code

```
DATA(lo_put_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-drty->add_object( '$CDS_TYPE_DEFINITION_NAME$'
    )->set_package( '$PACKAGE_NAME$'
    )->create_form_specification( ).

lo_form_specification->set_short_description( 'My short description' ).

DATA(lo_simple_type_definition) = lo_form_specification->add_simple_type_definition( ).
lo_simple_type_definition->add_annotation( 'EndUserText.label'
    )->value->build(
        )->add_string( 'My simple type definition' ).
lo_simple_type_definition->set_type( xco_cp_abap_dictionary->built_in_type->char( 30 ) ).

lo_put_operation->execute( ).
```

### DELETE Operations

CDS type definitions can be deleted via DELETE operations. As per the standard semantics of DELETE operations, if the CDS type definition for which a DELETE operation is being run doesn't exist, no action will be performed (and no error will be raised). If the CDS type definition does exist, it will be deleted. The code sample below illustrates how to run a DELETE operation for a CDS type definition:

#### Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->for-drty->create_delete_operation( ).

lo_delete_operation->add_object( '$CDS_TYPE_DEFINITION_NAME$' ).

lo_delete_operation->execute( ).
```

## 4.2.11.16.3.8.2 Data Definitions

Find out how to create data definitions.

Data definitions, also called objects of type `DDLS`, are supported by the following API families within the [XCO Library](#):

- Query APIs
- Read APIs
- Generation APIs

In addition to the standard API families listed above, dedicated APIs are offered for the following aspects:

- Setting and getting the API state of a CDS entity

### Query APIs

Obtaining an object handle for a data definition with a given name can be accomplished via

#### ↳ Sample Code

```
DATA(lo_data_definition) = xco_cp_abap_repository->object->ddls->for( '$DATA_DEFINITION_NAME$' ).
```

The obtained object handle can then be used to check the existence of the data definition in the ABAP Repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible data definitions in the ABAP Repository can simply be accomplished via

#### ↳ Sample Code

```
DATA(lt_data_definitions) = xco_cp_abap_repository->objects->ddls->all->in( xco_cp_abap=>repository )->get( ).  
LOOP AT lt_data_definitions ASSIGNING FIELD-SYMBOL(<fs_data_definition>).  
    " Process the found data definition.  
ENDLOOP.
```

Instead of searching for data definitions in the entire ABAP Repository, it's also possible to only search for data definitions in a dedicated subset of the ABAP Repository, such as a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).  
DATA(lt_data_definitions) = xco_cp_abap_repository->objects->ddls->all->in( lo_package )->get( ).  
LOOP AT lt_data_definitions ASSIGNING FIELD-SYMBOL(<fs_data_definition>).  
    " Process the found data definition.  
ENDLOOP.
```

Another way to refine a search for data definitions is to use filters to specify exactly which data definitions shall be matched, for example:

#### ↔ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(
    xco_cp_abap_sql=>constraint->contains_pattern( '/ABC/%' )
).

" LT_DATA_DEFINITIONS will be a list of all visible data definitions in the
ABAP
" Repository whose name starts with /ABC/ .
DATA(lt_data_definitions) = xco_cp_abap_repository->objects->ddls-
>where( VALUE #(
    ( lo_object_name_filter )
) )->in( xco_cp_abap=>repository )->get( ).

LOOP AT lt_data_definitions ASSIGNING FIELD-SYMBOL(<fs_data_definition>).
    " Process the found data definition.
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of a data definition is possible via its object handle `IF_XCO_DATA_DEFINITION`, respectively the corresponding interfaces for each supported data definition type:

- View entity: `IF_XCO_CDS_VIEW_ENTITY`
- Project view: `IF_XCO_CDS_PROJECTION_VIEW`
- Abstract entity: `IF_XCO_CDS_ABSTRACT_ENTITY`
- Custom entity: `IF_XCO_CDS_CUSTOM_ENTITY`
- Table function: `IF_XCO_CDS_TABLE_FUNCTION`

When reading the content of a CDS entity, the read state for which the content shall be read can explicitly be specified. The following read states are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_DATA_DEFINITION=>OBJECT_READ_STATE=ACTIVE`. The active read state is used by default, so if no read state is explicitly specified when reading the content of a data definition or checking its existence
- **LATEST**: Can be obtained via `XCO_CP_DATA_DEFINITION=>OBJECT_READ_STATE=LATEST`

Following the general style of the XCO Read APIs, each part of a CDS entity can be read out separately:

### View Entities

The following code sample illustrates how the (structured) content of a view entity can be read out:

#### ↔ Sample Code

```
DATA(lo_view_entity) = xco_cp_cds=>view_entity( '$VIEW_ENTITY_NAME$' ).

" The (structured) header content.
DATA(ls_header) = lo_view_entity->content( )->get( ).

" Read out the parameters of the view entity.
```

```

DATA(lt_parameters) = lo_view_entity->parameters->all->get( ).

LOOP AT lt_parameters ASSIGNING FIELD-SYMBOL(<fs_parameter>).
  " The (structured) content of the parameter.
  DATA(ls_parameter) = <fs_parameter>->content( )->get( ).
ENDLOOP.

" Read out the associations of the view entity.
DATA(lt_associations) = lo_view_entity->associations->all->get( ).

LOOP AT lt_associations ASSIGNING FIELD-SYMBOL(<fs_association>).
  " The (structured) content of the association.
  DATA(ls_association) = <fs_association>->content( )->get( ).
ENDLOOP.

" Read out the compositions of the view entity.
DATA(lt_compositions) = lo_view_entity->compositions->all->get( ).

LOOP AT lt_compositions ASSIGNING FIELD-SYMBOL(<fs_composition>).
  " The (structured) content of the composition.
  DATA(ls_composition) = <fs_composition>->content( )->get( ).
ENDLOOP.

" Read out the fields of the view entity.
DATA(lt_fields) = lo_view_entity->fields->all->get( ).

LOOP AT lt_fields ASSIGNING FIELD-SYMBOL(<fs_field>).
  " The (structured) content of the field.
  DATA(ls_field) = <fs_field>->content( )->get( ).
ENDLOOP.

```

## Projection Views

The following code sample illustrates how the (structured) content of a projection view can be read out:

### ↳ Sample Code

```

DATA(lo_projection_view) =
xco_cp_cds=>projection_view( '$PROJECTION_VIEW_NAME$' ).

" The (structured) header content.
DATA(ls_header) = lo_projection_view->content( )->get( ).

" Read out the fields of the projection view.
DATA(lt_fields) = lo_projection_view->fields->all->get( ).

LOOP AT lt_fields ASSIGNING FIELD-SYMBOL(<fs_field>).
  " The (structured) content of the field.
  DATA(ls_field) = <fs_field>->content( )->get( ).
ENDLOOP.

```

## Abstract Entities

The following code sample illustrates how the (structured) content of an abstract entity can be read out:

### ↳ Sample Code

```

DATA(lo_abstract_entity) =
xco_cp_cds=>abstract_entity( '$ABSTRACT_ENTITY_NAME$' ).

" The (structured) header content.
DATA(ls_header) = lo_abstract_entity->content( )->get( ).

```

```

" Read out the parameters of the abstract entity.
DATA(lt_parameters) = lo_abstract_entity->parameters->all->get( ).

LOOP AT lt_parameters ASSIGNING FIELD-SYMBOL(<fs_parameter>).
  " The (structured) content of the parameter.
  DATA(ls_parameter) = <fs_parameter>->content( )->get( ).
ENDLOOP.

" Read out the fields of the abstract entity.
DATA(lt_fields) = lo_abstract_entity->fields->all->get( ).

LOOP AT lt_fields ASSIGNING FIELD-SYMBOL(<fs_field>).
  " The (structured) content of the field.
  DATA(ls_field) = <fs_field>->content( )->get( ).
ENDLOOP.

```

## Custom Entities

The following code sample illustrates how the (structured) content of a custom entity can be read out:

### ↳ Sample Code

```

DATA(lo_custom_entity) = xco_cp_cds=>custom_entity( '$CUSTOM_ENTITY_NAME$' ).

" The (structured) header content.
DATA(ls_header) = lo_custom_entity->content( )->get( ).

" Read out the parameters of the custom entity.
DATA(lt_parameters) = lo_custom_entity->parameters->all->get( ).

LOOP AT lt_parameters ASSIGNING FIELD-SYMBOL(<fs_parameter>).
  " The (structured) content of the parameter.
  DATA(ls_parameter) = <fs_parameter>->content( )->get( ).
ENDLOOP.

" Read out the fields of the custom entity.
DATA(lt_fields) = lo_custom_entity->fields->all->get( ).

LOOP AT lt_fields ASSIGNING FIELD-SYMBOL(<fs_field>).
  " The (structured) content of the field.
  DATA(ls_field) = <fs_field>->content( )->get( ).
ENDLOOP.

```

## Table Functions

The following code sample illustrates how the (structured) content of a table function can be read out:

### ↳ Sample Code

```

DATA(lo_table_function) =
xco_cp_cds=>table_function( '$TABLE_FUNCTION_NAME$' ).

" The (structured) header content.
DATA(ls_header) = lo_table_function->content( )->get( ).

" Read out the parameters of the table_function.
DATA(lt_parameters) = lo_table_function->parameters->all->get( ).

LOOP AT lt_parameters ASSIGNING FIELD-SYMBOL(<fs_parameter>).
  " The (structured) content of the parameter.
  DATA(ls_parameter) = <fs_parameter>->content( )->get( ).
ENDLOOP.

```

```

" Read out the fields of the table function.
DATA(lt_fields) = lo_table_function->fields->all->get( ).

LOOP AT lt_fields ASSIGNING FIELD-SYMBOL(<fs_field>).
  " The (structured) content of the field.
  DATA(ls_field) = <fs_field>->content( )->get( ).
ENDLOOP.
```

## Generation APIs

Data definitions are integrated with XCO Generation APIs. The following operations are currently offered:

- **PUT**: Create or update a data definition based on a complete specification of its content, so that after the successful execution of the operation, the data definition exists and its content is exactly as specified. Both form specifications as well as source templates can be used to provide a complete content specification
- **DELETE**: Delete a data definition (if it exists)

### PUT Operations

Upon running the **PUT** operation, depending on whether the data definition already exists or not, it will either be newly created or updated. This is done so that after the successful completion of the **PUT** operation, the data definition exists with the content resembling exactly what was specified in the provided content specification. In addition to the traditional form specification, data definitions also support the usage of source templates to specify the source for the data definition directly.

#### ⓘ Note

Note that the data definition examples given below are intended exclusively to illustrate how the respective facilities of the XCO Generation APIs can be used to programmatically create (or update) data definitions. The examples are chosen in a way that syntactically error-free data definitions are generated and are not meant to be real-world modelling examples. In particular, note that certain examples, such as those for projection views, might not correlate with general best modelling practices.

### View Entity Form Specification

The content of a view entity can be specified via a form specification. The following code sample illustrates how a simple view entity (based on **I\_Language** for the purposes of this example) can be generated:

#### ↔ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST$'
 )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( '$VIEW_ENTITY_NAME$'
 )->set_package( '$PACKAGE_NAME$'
 )->create_form_specification( ).
lo_form_specification->set_short_description( 'Generated view entity' ).

DATA(lo_view_entity) = lo_form_specification->add_view_entity( ).
lo_view_entity->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My view entity| ).
lo_view_entity->set_root( ).
lo_view_entity->data_source->set_view_entity( 'I_Language' ).
```

```

DATA(lo_parameter) = lo_view_entity->add_parameter( 'myParameter' ).
lo_parameter->set_data_type( xco_cp_abap_dictionary=>built_in_type-
>char( 10 ) ).
lo_parameter->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My parameter| ).

DATA(lo_field) = lo_view_entity->add_field( xco_cp_ddl=>field( 'Language' )
    )->set_key(
    )->set_alias( 'myKeyField' ).
lo_field->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My key field| ).
lo_field->add_comment( xco_cp=>string( 'This is a key field' ) .
lo_put_operation->execute( ).
```

After successfully running the above `PUT` operation, a view entity with the name `$VIEW_ENTITY_NAME$` will exist in an active version with the following (source) content:

#### ↳ Sample Code

```

@EndUserText.label: 'My view entity'
define root view entity $VIEW_ENTITY_NAME$
with parameters
    @EndUserText.label: 'My parameter'
    myParameter : abap.char( 10 )
    as select from I_Language
{
    @EndUserText.label: 'My key field'
    key Language as myKeyField
}
```

#### Projection View Form Specification

The content of a projection view can be specified via a form specification. The following code sample illustrates how a simple projection view (based on `I_Language` and modelled as an analytical query for the purposes of this example) can be generated:

#### ↳ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( '$PROJECTION_VIEW_NAME$'
    )->set_package( '$PACKAGE_NAME$'
    )->create_form_specification( .
lo_form_specification->set_short_description( 'Generated projection view' ).

DATA(lo_projection_view) = lo_form_specification->add_projection_view( .
lo_projection_view->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My projection view| .
lo_projection_view->add_annotation( 'AccessControl.authorizationCheck' )-
>value->build( )->add_enum( 'NOT_ALLOWED' .
lo_projection_view->set_transient( .
lo_projection_view->set_provider_contract( xco_cp_cds=>provider_contract-
>analytical_query .
lo_projection_view->data_source->set_view_entity( 'I_Language' .

DATA(lo_field) = lo_projection_view-
>add_field( xco_cp_ddl=>field( 'Language' )
    )->set_alias( 'myField' ).
```

```

lo_field->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My field| ).

lo_put_operation->execute( ).
```

After the successful execution of the above PUT operation, a projection view with the name \$PROJECTION\_VIEW\_NAME\$ will exist in active version with the following (source) content:

#### ↔ Sample Code

```

@EndUserText.label: 'My projection view'
@AccessControl.authorizationCheck: #NOT_ALLOWED
define transient view entity $PROJECTION_VIEW_NAME$
  provider contract analytical_query
  as projection on I_Language
{
  @EndUserText.label: 'My field'
  Language as myField
}
```

### Abstract Entity Form Specification

The content of an abstract entity can be specified via a form specification. The following code sample illustrates how a simple abstract entity can be generated:

#### ↔ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( '$TRANSPORT_REQUEST$'
 )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( '$ABSTRACT_ENTITY_NAME$'
 )->set_package( '$PACKAGE_NAME$'
 )->create_form_specification( ).
lo_form_specification->set_short_description( 'Generated abstract entity' ).

DATA(lo_abstract_entity) = lo_form_specification->add_abstract_entity( ).
lo_abstract_entity->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My abstract entity| ).

DATA(lo_parameter) = lo_abstract_entity->add_parameter( 'MY_PARAMETER' ) .
lo_parameter->set_data_type( xco_cp_abap_dictionary=>built_in_type-
>char( 10 ) ) .
lo_parameter->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My parameter| ) .

DATA(lo_field) = lo_abstract_entity-
>add_field( xco_cp_ddl=>field( 'MY_FIELD' ) ) .
lo_field->set_type( xco_cp_abap_dictionary=>data_element( 'SPRAS' ) ) .
lo_field->add_annotation( 'EndUserText.label' )->value->build( )-
>add_string( |My field| ) .

lo_put_operation->execute( ).
```

After successfully running the above PUT operation, an abstract entity with the name \$ABSTRACT\_ENTITY\_NAME\$ will exist in an active version with the following (source) content:

## ↔ Sample Code

```
@EndUserText.label: 'My abstract entity'
define abstract entity $ABSTRACT_ENTITY_NAME$
  with parameters
    @EndUserText.label: 'My parameter'
    MY_PARAMETER : abap.char( 10 )
{
  @EndUserText.label: 'My field'
  MY_FIELD : SPRAS;
}
```

## Custom Entity Form Specification

The content of a custom entity can be specified via a form specification. The following code sample illustrates how a simple custom entity can be generated:

## ↔ Sample Code

```
DATA(lo_put_operation) = xco_cp_generation=>environment->dev_system( '$TRANSPORT_REQUEST$' )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls->add_object( '$CUSTOM_ENTITY_NAME$' )->set_package( '$PACKAGE_NAME$' )->create_form_specification( ).  
lo_form_specification->set_short_description( 'Generated custom entity' ).  
  
DATA(lo_custom_entity) = lo_form_specification->add_custom_entity( ).  
lo_custom_entity->add_annotation( 'EndUserText.label' )->value->build( )->add_string( |My custom entity| ).  
  
DATA(lo_parameter) = lo_custom_entity->add_parameter( 'myParameter' ).  
lo_parameter->set_data_type( xco_cp_abap_dictionary=>built_in_type->char( 10 ) ).  
lo_parameter->add_annotation( 'EndUserText.label' )->value->build( )->add_string( |My parameter| ).  
  
DATA(lo_key_field) = lo_custom_entity->add_field( xco_cp_ddl=>field( 'myKeyField' ) ).  
lo_key_field->set_key( ).  
lo_key_field->set_type( xco_cp_abap_dictionary=>built_in_type->char( 10 ) ).  
lo_key_field->add_annotation( 'EndUserText.label' )->value->build( )->add_string( |My key field| ).  
  
DATA(lo_data_field) = lo_custom_entity->add_field( xco_cp_ddl=>field( 'myDataField' ) ).  
lo_data_field->set_type( xco_cp_abap_dictionary=>data_element( 'MSEHI' ) ).  
lo_data_field->add_annotation( 'EndUserText.label' )->value->build( )->add_string( |My data field| ).  
  
lo_put_operation->execute( ).
```

After successful execution, a custom entity with the name `$CUSTOM_ENTITY_NAME$` will exist in an active version with the following (source) content:

## ↔ Sample Code

```
@EndUserText.label: 'My abstract entity'
define custom entity $CUSTOM_ENTITY_NAME$
```

```

with parameters
  @EndUserText.label: 'My parameter'
  myParameter : abap.char( 10 )
{
  @EndUserText.label: 'My key field'
  key myKeyField : abap.char( 10 );
  @EndUserText.label: 'My data field'
  myDataField : MSEHI;
}

```

### Table Function Form Specification

The content of a table function can be specified via a form specification. The following code sample illustrates how a simple table function can be generated:

#### Sample Code

```

DATA(lo_put_operation) = xco_cp_generation=>environment-
>dev_system( '$TRANSPORT_REQUEST$'
 )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ddls-
>add_object( '$TABLE_FUNCTION_NAME$'
 )->set_package( '$PACKAGE_NAME$'
 )->create_form_specification( ).
lo_form_specification->set_short_description( 'Generated table function' ).

DATA(lo_table_function) = lo_form_specification->add_table_function( ).
lo_table_function->add_annotation( 'AccessControl.authorizationCheck'
 )->value->build(
 )->add_enum( 'NOT_REQUIRED' ).
lo_table_function->add_annotation( 'ClientHandling.type'
 )->value->set( xco_cp_cds_annotation=>value->enum( 'CLIENT_INDEPENDENT' ) ).

DATA(lo_parameter) = lo_table_function->add_parameter( 'p_identifier'
 )->set_data_type( xco_cp_abap_dictionary=>data_element( 'syuname' ) ).

DATA(lo_field) = lo_table_function-
>add_field( xco_cp_ddl=>field( 'val_identifier' ) ).
lo_field->set_type( xco_cp_abap_dictionary=>built_in_type->char( 30 ) ).

lo_field->add_annotation( 'EndUserText.label'
 )->value->set( xco_cp_cds_annotation=>value->string( 'Identifier field' ) )
##NO_TEXT.

lo_table_function->set_amdp_class( '$AMDP_CLASS_NAME$'
 )->set_amdp_method( '$AMDP_CLASS_METHOD$' ).

lo_put_operation->execute( ).

```

After successful execution, a table function with the name `$TABLE_FUNCTION_NAME$` will exist in an active version with the following (source) content:

#### Sample Code

```

@AccessControl.authorizationCheck: #NOT_REQUIRED
@ClientHandling.type: #CLIENT_INDEPENDENT
define table function $TABLE_FUNCTION_NAME$
  with parameters
    p_identifier : syuname
  returns
{

```

```

        @EndUserText.label: 'Identifier field'
        val_identifier : abap.char( 30 );

    }
    implemented by method
    $AMDP_CLASS_NAME$=>$AMDP_METHOD_NAME$

```

## Source Templates

As an alternative to providing the content of a data definition via a form specification, it's possible to use a source template where the source of the data definition can be passed as a plain string. The following example illustrates how this can be used to generate a simple abstract entity by supplying its complete source directly:

### ↳ Sample Code

```

DATA(lo_put_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST$'
 )->create_put_operation( ).

DATA(lo_source_template) = xco_cp_generation_ddls->template->source( ).

DATA(lv_source) = xco_cp=>strings( VALUE #(
( `define abstract entity $ABSTRACT_ENTITY_NAME$` )
( `{` )
( ` // My field` )
( ` MY_FIELD : spras;` )
( `}` )
) )->join( |{ cl_abap_char_utilities=>cr_lf }| )->value.

lo_source_template->set_short_description( 'Generated abstract entity'
)->set_source( lv_source ).

lo_put_operation->for-ddls->add_object( '$ABSTRACT_ENTITY_NAME$'
)->set_package( '$PACKAGE_NAME$'
)->set_template( lo_source_template ).

lo_put_operation->execute( ).

```

Once the PUT operation has run successfully, an abstract entity with the name \$ABSTRACT\_ENTITY\_NAME\$ will exist in an active version with the following (source) content:

### ↳ Sample Code

```

define abstract entity $ABSTRACT_ENTITY_NAME$
{
  // My field
  MY_FIELD : spras;
}

```

## DELETE Operations

Data definitions (regardless of the type of entity they define) can be deleted via DDLS DELETE operations. As per the standard semantics of DELETE operations, if the data definition for which a DELETE operation is run doesn't exist, no action will be performed (and no error will be raised). If the data definition does exist, it will be deleted. The code sample below illustrates how to execute a DELETE operation for a data definition:

### ↳ Sample Code

```

DATA(lo_delete_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST'

```

```

) -> for-ddls->create_delete_operation( ).

lo_delete_operation->add_object( '$DATA_DEFINITION_NAME$' ).

lo_delete_operation->execute( ).

```

## Setting and Getting the API State of a CDS Entity

A CDS entity defined by a data definition can be released via the API state mechanisms of the [API Release Framework](#) (ARS). The [XCO Library](#) allows to programmatically set and get the API state of a CDS entity (represented as `IF_XCO_CDS_ENTITY`). The following code sample illustrates how the API state of a given CDS entity can be read:

### Sample Code

```

DATA(lo_cds_entity) = xco_cp_cds=>entity( '$CDS_ENTITY_NAME$' ).

" Get the (current) API state of the CDS entity for the C1 release contract.
DATA(lo_api_state) = lo_cds_entity-
>get_api_state( xco_cp_ars=>release_contract->c1 ).

" The returned LO_API_STATE object allows to inspect the individual attributes
" of the API state.
DATA(lo_release_state) = lo_api_state->get_release_state( ).

" Release states supported by the XCO Library can be obtained via
XCO_CP_ARS=>RELEASE_STATE.
" Checking whether the API state of the CDS entity has the release state
" "RELEASED"
" would look like this:
IF lo_release_state EQ xco_cp_ars=>release_state->released.
" ...
ENDIF.

DATA(lt_visibilities) = lo_api_state->get_visibilities( ).

" Visibilities supported by the XCO Library can be obtained via
XCO_CP_ARS=>VISIBILITY.
" Note that due to compatibility and historical reasons, the visibility
" CLOUD_DEVELOPMENT is called SAP_CLOUD_PLATFORM within the XCO Library.
Checking
" whether the visibilities of the API state include CLOUD_DEVELOPMENT would
" then look like this:
IF line_exists( lt_visibilities[ table_line = xco_cp_ars=>visibility-
>sap_cloud_platform ] ).
" ...
ENDIF.

```

Setting the API state for a CDS entity can be done like this:

### Sample Code

```

" The API state which shall be set for the CDS entity is built via
XCO_CP_ARS=>API_STATE:
DATA(lo_api_state) = xco_cp_ars=>api_state->released( VALUE
#( ( xco_cp_ars=>visibility->sap_cloud_platform ) ) ).

" Setting the API state for a CDS entity implies a change which must be
recorded

```

```

" on an appropriate transport.
DATA(lo_transport) = xco_cp_cts->transport->for( '$TRANSPORT$' ).

" The API state is set via the handle for the CDS entity.
DATA(lo_cds_entity) = xco_cp_cds->entity( '$CDS_ENTITY_NAME$' ).

" We set the previously built API State (Released for Cloud Development) using
" the C1-release contract for the CDS entity (recording the changes on the
" provided transport).
lo_cds_entity->set_api_state(
    io_release_contract = xco_cp_ars->release_contract->c1
    io_change_scenario  = lo_transport
    io_api_state         = lo_api_state
).

```

### 4.2.11.16.3.8.3 Domains

Find out how to create domains.

Domains, meaning objects of type DOMA, are supported by the following API families within the XCO library:

- Query APIs
- Read APIs
- Generation APIs

In addition to the standard API families listed above, dedicated APIs are offered for the following aspects:

- Setting and getting the API state of a domain

#### Query APIs

Obtaining an object handle for a domain with a given name can be accomplished via

##### ↳ Sample Code

```
DATA(lo_domain) = xco_cp_abap_repository->object->doma-
>for( '$DOMAIN_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the domain in the ABAP repository (via method IF\_XCO\_AR\_OBJECT~EXISTS or read out its content via the Read APIs (see below)). Obtaining a list of all visible domains in the ABAP repository can be done via

##### ↳ Sample Code

```
DATA(lt_domains) = xco_cp_abap_repository->objects->doma->all-
>in( xco_cp_abap->repository )->get( ).

LOOP AT lt_domains ASSIGNING FIELD-SYMBOL(<fs_domain>).
    " Process the found domain.
ENDLOOP.
```

Instead of searching for domains in the entire ABAP repository, it's also possible to only search for domains in a dedicated subset of the ABAP repository, such as in a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ) .  
DATA(lt_domains) = xco_cp_abap_repository->objects->doma->all-  
>in( lo_package )->get( ).  
LOOP AT lt_domains ASSIGNING FIELD-SYMBOL(<fs_domain>).  
    " Process the found domain.  
ENDLOOP.
```

Furthermore, you can refine a search for domains by using filters to specify exactly which domains should be matched, for example:

#### ↳ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(  
    xco_cp_abap_sql->constraint->contains_pattern( '/ABC/%' )  
).  
" LT_DOMAINS will be a list of all visible domains in the ABAP Repository  
" whose name starts with /ABC/.  
DATA(lt_domains) = xco_cp_abap_repository->objects->doma->where( VALUE #(  
    ( lo_object_name_filter )  
) )->in( xco_cp_abap->repository )->get( ).  
LOOP AT lt_domains ASSIGNING FIELD-SYMBOL(<fs_domain>).  
    " Process the found domain.  
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of a domain is possible via its object handle `IF_XCO_DOMAIN`. The Read APIs for domains are based on the origin infrastructure with the following origins being offered currently for domains:

- **DEFAULT**: Can be obtained via `XCO_CP_DOMAIN=>ORIGIN=>DEFAULT` and will read a given domain exactly once per ABAP session from the local system, with all subsequent reads of the domain being fulfilled from a dedicated buffer. This origin is used by default, meaning if no origin is explicitly specified when reading the content or checking the existence of a domain.
- **LOCAL**: Can be obtained via `XCO_CP_DOMAIN=>ORIGIN=>LOCAL` and will read a given domain from the local system. By default, every read of the content as well as existence checks of a domain will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_DOMAIN_CACHEABLE_ORIGIN`.

Next, when reading a domain or checking its existence, it can explicitly be specified which read state of the domain should be considered. The following read states are supported:

- **ACTIVE VERSION**: Can be obtained via `XCO_CP_ABAP_TYPE_DICTIONARY=>OBJECT_READ_STATE=>ACTIVE_VERSION`. The active version is

used by default, meaning if no read state is explicitly specified when reading the content of a domain or checking its existence.

- **NEWEST VERSION:** Can be obtained via  
XCO\_CP\_ABAP\_DICTIONARY=>OBJECT\_READ\_STATE->NEWEST\_VERSION.

Following the general style of the XCO Read APIs, each part of a domain can be read out separately:

## Header

Reading out the header of a domain can be done like this:

### Sample Code

```
DATA(lo_domain) = xco_cp_abap_repository=>object->doma-
>for( '$DOMAIN_NAME$' ).

" Read out the complete header of the active version of the domain using the
" default origin:
DATA(ls_header) = lo_domain->content( )->get( ).

" Read out only the short description of the newest version of the domain
" using an explicitly obtained local origin (without caching).
DATA(lo_local_origin) = xco_cp_domain=>origin->local( ).

DATA(lv_short_description) = lo_domain-
>content( xco_cp_abap_dictionary=>object_read_state->newest_version
 )->get_short_description( lo_local_origin ).
```

## Fixed Values

Obtaining a list of the fixed values of a domain and reading out an individual fixed value can be done like this:

### Sample Code

```
DATA(lo_domain) = xco_cp_abap_repository=>object->doma-
>for( '$DOMAIN_NAME$' ).

" Get a list of all the fixed values of the active version of the domain
" using the default origin:
DATA(lt_fixed_values) = lo_domain->fixed_values->all->get( ).

LOOP AT lt_fixed_values ASSIGNING FIELD-SYMBOL(<fs_fixed_value>).
  " Process the fixed value.
ENDLOOP.

" Obtain the handle for an individual fixed value.
DATA(lo_fixed_value) = lo_domain->fixed_value( 'A' ).

" Read out the complete content of the active version of the fixed value
" using the default origin.
DATA(ls_fixed_value) = lo_fixed_value->content( )->get( ).

" Read out only the description of the newest version of the fixed value
" using an explicitly obtained local origin with caching enabled.
DATA(lo_local_origin) = xco_cp_domain=>origin->local( ).
lo_local_origin->if_xco_domain_cacheable_origin~enable_caching( ).

DATA(lv_description) = lo_fixed_value-
>content( xco_cp_abap_dictionary=>object_read_state->newest_version
 )->get_description( lo_local_origin ).
```

## Generation APIs

Domains are integrated with the XCO Generation APIs with the following operations being offered:

- **PUT**: Create or update a domain based on an entire content specification so that after running the operation successfully, the domain exists and its content is exactly as specified
- **PATCH**: Change an existing domain by applying the changes which have been provided in a change specification
- **DELETE**: Delete a domain (if it exists)

### PUT Operations

Upon running a PUT operation, depending on whether the domain already exists or not, it will either be newly created or updated. This is so that after the successful completion of the PUT operation, the domain exists with the content resembling exactly what was specified in the provided content specification. In addition to the traditional form specification, domains also support the use of object templates to use the content of an existing domain as the content for the domain of the PUT operation.

#### Form specification

Consider the following code sample which illustrates how to run a PUT operation for a domain using a form specification:

#### Sample Code

```
DATA(lo_put_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for_domain->add_object( '$DOMAIN_NAME$'
    )->set_package( '$PACKAGE_NAME$'
    )->create_form_specification( ).

lo_form_specification->set_short_description( 'My short description'
    )->set_format( xco_cp_abap_dictionary->built_in_type->numc( 2 ) ).

DATA(lo_fixed_value_01) = lo_form_specification->fixed_values->add_fixed_value( '01' ).
lo_fixed_value_01->set_description( 'January' ).

lo_put_operation->execute( ).
```

#### Object template

An object template can be used to effectively copy an already existing domain to a new domain. The following code sample illustrates the use:

#### Sample Code

```
DATA(lo_put_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_object_template) = xco_cp_generation_domain->template->for_domain(
    iv_name      = '$SOURCE_DOMAIN_NAME$'
    io_read_state = xco_cp_abap_dictionary->object_read_state->active_version
).
```

```

lo_put_operation->for-doma->add_object( '$DOMAIN_NAME$'
)->set_package( '$PACKAGE_NAME$'
)->set_template( lo_object_template ).

lo_put_operation->execute( ).
```

## PATCH Operations

A change specification is used to specify the changes that should be applied to an existing domain, and is the basis for running a PATCH operation for a domain. Upon running a PATCH operation, the changes will be applied to the domain. The following kinds of changes can be specified:

- **DELETE**: If the specified part exists in the domain, it'll be deleted. If it doesn't exist, no action will be performed.
- **INSERT**: If the specified part doesn't exist in the domain, it'll be inserted according to the provided specification. If it exists, no action will be performed.
- **UPDATE**: If the specified part exists in the domain, it'll be updated according to the provided specification.

Note that the different kinds of changes are always evaluated in the order Delete, Insert, Update with the changes that are applied for one kind being directly visible to the changes that are applied for a subsequent kind. This orchestration can be exploited to, for example, implement a MODIFY by specifying both a DELETE and an INSERT for a given object part. The following code example illustrates selected changes and how they can be applied via a (single) PATCH operation:

### Sample Code

```

DATA(lo_patch_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST$'
)->create_patch_operation( ).

DATA(lo_change_specification) = lo_patch_operation->for-doma-
>add_object( '$DOMAIN_NAME$'
)->create_change_specification( ).

" Update the short description.
lo_change_specification->for-update->set_short_description( 'New short
description' ).

" Modify fixed version 02 by specifying both a deletion and an insertion for
" it.
lo_change_specification->for-delete->add_fixed_value( '02' ).

DATA(lo_fixed_value_02) = lo_change_specification->for-insert-
>add_fixed_value( '02' ).
lo_fixed_value_02->set_description( 'February' ).

" Update the description of fixed value 01 (if it exists in the domain).
lo_change_specification->for-update->add_fixed_value( '01'
)->set_description( 'January' ).

" Delete fixed value 03 (if it exists in the domain).
lo_change_specification->for-delete->add_fixed_value( '03' ).

lo_patch_operation->execute( ).
```

## DELETE Operations

Domains can be deleted via DELETE operations. As per the standard semantics of DELETE operations, if the domain for which a DELETE operation is run doesn't exist, no action will be performed (and no error will

be raised). If the domain exists, it will be deleted. The code sample below illustrates how to run a DELETE operation for a domain:

#### ↳ Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation=>environment->dev_system( '$TRANSPORT_REQUEST$'
    )->for-domains->create_delete_operation( ).

lo_delete_operation->add_object( '$DOMAIN_NAME$' ).

lo_delete_operation->execute( ).
```

## Setting and Getting the API State of a Domain

A domain can be released via the API state mechanisms of the API Release Framework (ARS). The XCO library allows you to programmatically set and get the API state of a domain (represented as `IF_XCO_DOMAIN`). The following code sample illustrates how the API state of a given domain can be read:

#### ↳ Sample Code

```
DATA(lo_domain) = xco_cp_abap_dictionary=>domain( '$DOMAIN_NAME$' ).

" Get the (current) API state of the domain for the C1 release contract.
DATA(lo_api_state) = lo_domain->get_api_state( ).

" The returned LO_API_STATE object allows to inspect the individual
" attributes of the API state.
DATA(lo_release_state) = lo_api_state->get_release_state( ).

" Release states supported by the XCO Library can be obtained via
XCO_CP_ARS=>RELEASE_STATE.
" Checking whether the API state of the domain has the release state
" "RELEASED" would look like this:
IF lo_release_state EQ xco_cp_ars=>release_state->released.
" ...
ENDIF.

DATA(lt_visibilities) = lo_api_state->get_visibilities( ).

" Visibilities supported by the XCO Library can be obtained via
XCO_CP_ARS=>VISIBILITY.
" Note that due to compatibility and historical reasons, the visibility
" CLOUD_DEVELOPMENT is called SAP_CLOUD_PLATFORM within the XCO Library.
Checking
" whether the visibilities of the API state include CLOUD_DEVELOPMENT would
" then look like this:
IF line_exists( lt_visibilities[ table_line = xco_cp_ars=>visibility-
>sap_cloud_platform ] ).
" ...
ENDIF.
```

Setting the API state for a domain can be done like this:

#### ↳ Sample Code

```
" The API state which should be set for the domain is built via
XCO_CP_ARS=>API_STATE:
```

```

DATA(lo_api_state) = xco_cp_ars->api_state->released( VALUE
#( ( xco_cp_ars->visibility->sap_cloud_platform ) ) ).

" Setting the API state for a domain implies a change which must be recorded
" on an appropriate transport.
DATA(lo_transport) = xco_cp_cts->transport->for( '$TRANSPORT$' ).

" The API state is set via the handle for the domain.
DATA(lo_domain) = xco_cp_abap_dictionary->domain( '$DOMAIN_NAME$' ).

" We set the previously built API State (Released for Cloud Development)
" using the C1-release contract for the domain (recording the changes on the
" provided transport).
lo_domain->set_api_state(
  io_change_scenario = lo_transport
  io_api_state       = lo_api_state
).

```

## 4.2.11.16.3.8.4 Event Bindings

Find out how to create event bindings.

Event bindings, also called objects of type `EVTB`, are supported by the following API families within the [XCO Library](#):

- Query APIs
- Read APIs
- Generation APIs

### Query APIs

Obtaining an object handle for an event binding with a given name can be accomplished via

#### ↳ Sample Code

```

DATA(lo_event_binding) = xco_cp_abap_repository->object->evtb-
>for( '$EVENT_BINDING_NAME$' ).

```

The obtained object handle can then be used to check the existence of the event binding in the ABAP Repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible event bindings in the ABAP Repository can simply be accomplished via

#### ↳ Sample Code

```

DATA(lt_event_bindings) = xco_cp_abap_repository->objects->evtb->all-
>in( xco_cp_abap->repository )->get( ).

LOOP AT lt_event_bindings ASSIGNING FIELD-SYMBOL(<fs_event_binding>).
  " Process the found event binding.
ENDLOOP.

```

Instead of searching the entire ABAP Repository for service bindings, it's also possible to only search for data definitions in a dedicated subset of the ABAP Repository, such as a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).  
DATA(lt_event_bindings) = xco_cp_abap_repository->objects->evtb->all-  
>in( lo_package )->get( ).  
  
LOOP AT lt_event_bindings ASSIGNING FIELD-SYMBOL(<fs_event_binding>).  
    " Process the found event binding.  
ENDLOOP.
```

Another way to refine a search for event bindings is to use filters to specify exactly which event bindings shall be matched, for example:

#### ↳ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(  
    xco_cp_abap_sql->constraint->contains_pattern( '/ABC/%' )  
).  
  
" LT_EVENT_BINDINGS will be a list of all visible event bindings in the ABAP  
" Repository whose name starts with /ABC/.  
DATA(lt_event_bindings) = xco_cp_abap_repository->objects->evtb->where( VALUE  
#(  
    ( lo_object_name_filter )  
) ->in( xco_cp_abap->repository )->get( ).  
  
LOOP AT lt_event_bindings ASSIGNING FIELD-SYMBOL(<fs_event_binding>).  
    " Process the found event binding.  
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of an event binding is possible via its object handle `IF_XCO_EVENT_BINDING`. The read APIs for event bindings are based on the origin infrastructure with the following origins being currently offered for event bindings:

- **DEFAULT**: Can be obtained via `XCO_CP_EVENT_BINDING->ORIGIN->DEFAULT` and will read a given event binding exactly once per ABAP session from the local system with all subsequent reads of the event binding being fulfilled from a dedicated buffer. This origin is used by default if no origin is explicitly specified when reading the content or checking the existence of an event binding.
- **LOCAL**: Can be obtained via `XCO_CP_EVENT_BINDING->ORIGIN->LOCAL` and will read a given event binding from the local system. By default, every read of the content as well as existence check of an event binding will read directly from the database but the caching behavior can be configured via the methods of `IF_XCO_EB_CACHEABLE_ORIGIN`.

Furthermore, when reading an event binding or checking its existence, it can explicitly be specified which version of the event binding shall be considered. The following versions are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_EVENT_BINDING=>VERSION=ACTIVE`. The active version is used by default, if no version is explicitly specified when reading the content of an event binding or checking its existence.
- **INACTIVE**: Can be obtained via `XCO_CP_EVENT_BINDING=>VERSION=INACTIVE`.

Following the general style of the XCO Read APIs, each part of an event binding can be read out separately:

## Header

The following code sample illustrates how the header of an event binding can be read out:

### Sample Code

```
DATA(lo_event_binding) = xco_cp_abap_repository=>object->evtb->for( '$EVENT_BINDING_NAME$' ).  
  
" Read out the complete header of the active version of the event binding  
" using the default origin:  
DATA(ls_header) = lo_event_binding->content( )->get( ).  
  
" Read out only the short description of the inactive version of the event  
" binding using an explicitly obtained local origin (without caching).  
DATA(lo_local_origin) = xco_cp_event_binding=>origin->local( ).  
  
DATA(lv_short_description) = lo_event_binding->content( xco_cp_event_binding=>version->inactive  
)->get_short_description( lo_local_origin ).
```

## Event Versions

Obtaining a list of the event versions of an event binding and reading out an individual event version can be done like this:

```
DATA(lo_event_binding) = xco_cp_abap_repository=>object->evtb->for( '$EVENT_BINDING_NAME$' ).  
  
" Get a list of all the event versions of the active version of the event  
" binding using the default origin:  
DATA(lt_event_versions) = lo_event_binding->event_versions->all->get( ).  
  
LOOP AT lt_event_versions ASSIGNING FIELD-SYMBOL(<fs_event_version>).  
    " Process the event version.  
ENDLOOP.  
  
" Obtain the handle for an individual event version.  
DATA(lo_event_version) = lo_event_binding->event_version( 0001 ).  
  
" Read out the complete content of the active version of the event version  
" using the default origin.  
DATA(ls_event_version) = lo_event_version->content( )->get( ).  
  
" Read out only the minor version of the inactive version of the event  
" version using an explicitly obtained local origin with caching enabled.  
DATA(lo_local_origin) = xco_cp_event_binding=>origin->local( ).  
lo_local_origin->if_xco_eb_cacheable_origin~enable_caching( ).  
  
DATA(lv_minor_version) = lo_event_version->content( xco_cp_event_binding=>version->inactive  
)->get_minor_version( lo_local_origin ).
```

## Generation APIs

Event bindings are integrated with XCO Generation APIs. The following operations are currently offered:

- **PUT**: Create or update an event binding based on a complete specification of its content, so that after the successful execution of the operation, the event binding exists and its content is exactly as specified
- **PATCH**: Change an existing event binding by applying the changes which have been provided in a change specification
- **DELETE**: Delete an event binding (if it exists)

### PUT operations

A form specification is used to specify the complete content of an event binding which is the basis for executing a PUT operation. Upon execution of the PUT operation, depending on whether the event binding already exists or not, it will either be newly created or updated. After the successful completion of the PUT operation the event binding exists with the content resembling exactly what was specified in the provided form specification. Consider the following code sample which illustrates how to execute a PUT operation for an event binding:

#### ↳ Sample Code

```
DATA(lo_put_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-evtb->add_object( '$EVENT_BINDING_NAME$'
    )->set_package( '$PACKAGE_NAME$'
    )->create_form_specification( ).

lo_form_specification->set_short_description( 'My short description'
    )->set_type_namespace( '$TYPE_NS$'
    )->set_sap_object_type( '$SAP_OBJECT_TYPE$'
    )->set_operation( '$OPERATION' ).

DATA(lo_event_version_0001) = lo_form_specification->add_event_version( 0001 ).
lo_event_version_0001->set_entity_name( '$ENTITY_NAME$'
    )->set_entity_event_name( '$ENTITY_EVENT_NAME$' ).

lo_put_operation->execute( ).
```

### PATCH operations

A change specification is used to specify the changes that shall be applied to an existing event binding and is the basis for executing a PATCH operation for an event binding. Upon execution of a PATCH operation, the changes will be applied to the event binding. The following kinds of changes can be specified:

- **DELETE**: If the specified part exists in the event binding it will be deleted. If it doesn't exist no action will be performed.
- **INSERT**: If the specified part doesn't exist in the event binding it will be inserted according to the provided specification. If it exists no action will be performed.
- **UPDATE**: If the specified part exists in the event binding it will be updated according to the provided specification.

#### ⓘ Note

Note that the different kinds of changes are always evaluated in the order delete, insert, update with the changes that are applied for one kind being directly visible to the changes that are applied for a subsequent

kind. This orchestration can be exploited to, for example, implement a **MODIFY** by specifying both a **DELETE** and an **INSERT** for a given object part. The following code example illustrates selected changes and how they can be applied via a (single) **PATCH** operation

```
DATA(lo_patch_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_patch_operation( ).

DATA(lo_change_specification) = lo_patch_operation->for-evtb->add_object( '$EVENT_BINDING_NAME$'
    )->create_change_specification( ).

" Update the short description.
lo_change_specification->for-update->set_short_description( 'New short
description' ).

" Modify event version 0001 by specifying both a deletion and an insertion
" for it.
lo_change_specification->for-delete->add_event_version( 0001 ).

DATA(lo_event_version_0001) = lo_change_specification->for-insert->add_event_version( 0001 ).
lo_event_version_0001->set_minor_version( 0000 ).
lo_event_version_0001->set_patch_version( 0000 ).
lo_event_version_0001->set_entity_name( '$ENTITY_NAME' ).
lo_event_version_0001->set_entity_event_name( '$ENTITY_EVENT_NAME$' ).

" Update the Entity Event Name of event version 0002 (if it exists in the
" event binding).
lo_change_specification->for-update->add_event_version( 0002
    )->set_entity_event_name( 'CREATED2' ).

" Delete event version 0003 (if it exists in the event binding).
lo_change_specification->for-delete->add_event_version( 0003 ).

lo_patch_operation->execute( ).
```

## DELETE operations

Event bindings can be deleted via **DELETE** operations. As per the standard semantics of **DELETE** operations, if the data definition for which a **DELETE** operation is run doesn't exist, no action will be performed (and no error will be raised). If the data definition does exist, it will be deleted. The code sample below illustrates how to execute a **DELETE** operation for a data definition:

### ↔ Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->for-evtb->create_delete_operation( ).

lo_delete_operation->add_object( '$EVENT_BINDING_NAME$' ).

lo_delete_operation->execute( ).
```

## 4.2.11.16.3.8.5 SAP Object Node Types

Find out how to create SAP object node types.

SAP object node types, meaning objects of type NONT, are supported by the following API families within the XCO library:

- Query APIs
- Read APIs
- Generation APIs

### Query APIs

Obtaining an object handle for an SAP object node type with a given name can be accomplished via

#### ↳ Sample Code

```
DATA(lo_sap_object_node_type) = xco_cp_abap_repository=>object->nont->for( '$SAP_OBJECT_NODE_TYPE_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the SAP object node type in the ABAP repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible SAP object node types in the ABAP repository can be done via

#### ↳ Sample Code

```
DATA(lt_sap_object_node_types) = xco_cp_abap_repository=>objects->nont->all->in( xco_cp_abap=>repository )->get( ).  
  
LOOP AT lt_sap_object_node_types ASSIGNING FIELD-SYMBOL(<fs_sap_object_node_type>).  
    " Process the found SAP object node type.  
ENDLOOP.
```

Instead of searching for SAP object node types in the entire ABAP repository, it's also possible to only search for SAP object node types in a dedicated subset of the ABAP repository, such as in a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository=>package->for( '$PACKAGE_NAME$' ).  
  
DATA(lt_sap_object_node_types) = xco_cp_abap_repository=>objects->nont->all->in( lo_package )->get( ).  
  
LOOP AT lt_sap_object_node_types ASSIGNING FIELD-SYMBOL(<fs_sap_object_node_type>).  
    " Process the found SAP object node type.  
ENDLOOP.
```

Furthermore, you can refine a search for SAO object node types by using filters to specify exactly which SAP object node types should be matched, for example:

## ↔ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(
    xco_cp_abap_sql=>constraint->contains_pattern( '/ABC/%' )
).

" LT_SAP_OBJECT_NODE_TYPES will be a list of all visible SAP object node
" types in the ABAP Repository whose name starts with /ABC/.
DATA(lt Sap_object_node_types) = xco_cp_abap_repository->objects->nont-
>where( VALUE #((
    lo_object_name_filter
) )->in( xco_cp_abap=>repository )->get( ).

LOOP AT lt Sap_object_node_types ASSIGNING FIELD-
SYMBOL(<fs Sap_object_node_type>).
    " Process the found SAP object node type.
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of an SAP object node type is possible via its object handle `IF_XCO_SAP_OBJECT_NODE_TYPE`. The Read APIs for SAP object node types are based on the origin infrastructure with the following origins being offered currently for SAP object node types:

- **DEFAULT**: Can be obtained via `XCO_CP_SAP_OBJECT_NODE_TYPE=>ORIGIN=DEFAULT` and will read a given SAP object node type exactly once per ABAP session from the local system, with all subsequent reads of the SAP object node type being fulfilled from a dedicated buffer. This origin is used by default, meaning if no origin is explicitly specified when reading the content or checking the existence of an SAP object node type.
- **LOCAL**: Can be obtained via `XCO_CP_SAP_OBJECT_NODE_TYPE=>ORIGIN=LOCAL` and will read a given SAP object node type from the local system. By default, every read of the content as well as existence checks of an SAP object node type will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_NONT_CACHEABLE_ORIGIN`.

Next, when reading an SAP object node type or checking its existence, it can explicitly be specified which version of the SAP object node type should be considered. The following versions are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_SAP_OBJECT_NODE_TYPE=>VERSION=ACTIVE`. The active version is used by default, meaning if no version is explicitly specified when reading the content of an SAP object node type or checking its existence.
- **INACTIVE**: Can be obtained via `XCO_CP_SAP_OBJECT_NODE_TYPE=>VERSION=INACTIVE`.

Following the general style of the XCO Read APIs, each part of an SAP object node type can be read out separately:

### Header

Reading out the header of an SAP object node type can be done like this:

## ↔ Sample Code

```
DATA(lo_sap_object_node_type) = xco_cp_abap_repository->object->nont-
>for( '$SAP_OBJECT_NODE_TYPE_NAME$' ).
```

```

" Read out the complete header of the active version of the SAP object node
" type using the default origin:
DATA(ls_header) = lo_sap_object_node_type->content( )->get( ).

" Read out only the short description of the inactive version of the SAP
" object node type using an explicitly obtained local origin (without
" caching).
DATA(lo_local_origin) = xco_cp_sap_object_node_type->origin->local( ).

DATA(lv_short_description) = lo_sap_object_node_type-
>content( xco_cp_sap_object_node_type->version->inactive
)->get_short_description( lo_local_origin ).
```

## Generation APIs

SAP object node types are integrated with the XCO Generation APIs with the following operations being offered currently for SAP object node types:

- **PUT**: Create or update an SAP object node type based on an entire content specification so that after running the operation successfully, the SAP object node type exists and its content is exactly as specified
- **PATCH**: Change an existing SAP object node type by applying the changes which have been provided in a change specification
- **DELETE**: Delete an SAP object node type (if it exists)

### PUT Operations

A form specification is used to specify the entire content of an SAP object node type, which is the basis for running a PUT operation. Upon running the PUT operation, depending on whether the SAP object node type already exists or not, it will either be newly created or updated. This is so that after the successful completion of the PUT operation, the SAP object node type exists with the content resembling exactly what was specified in the provided form specification. Consider the following code sample which illustrates how to run a PUT operation for an SAP object node type:

#### Sample Code

```

DATA(lo_put_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST$'
)->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-nont-
>add_object( '$SAP_OBJECT_NODE_TYPE_NAME$'
)->set_package( '$PACKAGE_NAME$'
)->create_form_specification( ).

lo_form_specification->set_short_description( 'My short description'
)->set_name( '$sapObjectNodeTypeNames$'
)->set_sap_object_type( '$SAP_OBJECT_TYPE_NAME$'
)->set_root_node( abap_true ).

lo_put_operation->execute( ).
```

### PATCH Operations

A change specification is used to specify the changes that should be applied to an existing SAP object node type, and is the basis for running a PATCH operation for an SAP object node type. Upon running a PATCH

operation, the changes will be applied to the SAP object node type. The following code example illustrates selected changes and how they can be applied via a PATCH operation:

#### ↔ Sample Code

```
DATA(lo_patch_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_patch_operation( ).

DATA(lo_change_specification) = lo_patch_operation->for-nont->add_object( '$SAP_OBJECT_NODE_TYPE_NAME$'
    )->create_change_specification( ).

" Update the short description.
lo_change_specification->for-update->set_short_description( 'New short
description' ).

lo_patch_operation->execute( ).
```

## DELETE Operations

SAP object node types can be deleted via DELETE operations. As per the standard semantics of DELETE operations, if the SAP object node type for which a DELETE operation is being run doesn't exist, no action will be performed (and no error will be raised). If the SAP object node type exists, it will be deleted. The code sample below illustrates how to run a DELETE operation for an SAP object node type:

#### ↔ Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$'
    )->for-nont->create_delete_operation( ).

lo_delete_operation->add_object( '$SAP_OBJECT_NODE_TYPE_NAME$' ) .

lo_delete_operation->execute( ).
```

## 4.2.11.16.3.8.6 SAP Object Types

Find out how to create SAP object types.

SAP object types, meaning objects of type RONT, are supported by the following API families within the XCO library:

- Query APIs
- Read APIs
- Generation APIs

## Query APIs

Obtaining an object handle for an SAP object type with a given name can be accomplished via

#### ↳ Sample Code

```
DATA(lo_sap_object_type) = xco_cp_abap_repository->object->ront->for( '$SAP_OBJECT_TYPE_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the SAP object type in the ABAP repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible SAP object types in the ABAP repository can be done via

#### ↳ Sample Code

```
DATA(lt_sap_object_types) = xco_cp_abap_repository->objects->ront->all->in( xco_cp_abap=>repository )->get( ).  
LOOP AT lt_sap_object_types ASSIGNING FIELD-SYMBOL(<fs_sap_object_type>).  
    " Process the found SAP object type.  
ENDLOOP.
```

Instead of searching for SAP object types in the entire ABAP repository, it's also possible to only search for SAP object types in a dedicated subset of the ABAP repository, such as in a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).  
DATA(lt_sap_object_types) = xco_cp_abap_repository->objects->ront->all->in( lo_package )->get( ).  
LOOP AT lt_sap_object_types ASSIGNING FIELD-SYMBOL(<fs_sap_object_type>).  
    " Process the found SAP object type.  
ENDLOOP.
```

Furthermore, you can refine a search for SAP object types by using filters to specify exactly which SAP object types should be matched, for example:

#### ↳ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(  
    xco_cp_abap_sql=>constraint->contains_pattern( '/ABC/%' )  
).  
    " LT_SAP_OBJECT_TYPES will be a list of all visible SAP object types in the  
    " ABAP Repository whose name starts with /ABC/.  
DATA(lt_sap_object_types) = xco_cp_abap_repository->objects->ront->where( VALUE #(  
    ( lo_object_name_filter )  
) )->in( xco_cp_abap=>repository )->get( ).  
LOOP AT lt_sap_object_types ASSIGNING FIELD-SYMBOL(<fs_sap_object_type>).  
    " Process the found SAP object type.  
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of an SAP object type is possible via its object handle `IF_XCO_SAP_OBJECT_TYPE`. The Read APIs for SAP object types are based on the origin infrastructure with the following origins being offered currently for SAP object types:

- **DEFAULT**: Can be obtained via `xco_cp_sap_object_type=>origin->default` and will read a given SAP object type exactly once per ABAP session from the local system, with all subsequent reads of the SAP object type being fulfilled from a dedicated buffer. This origin is used by default, meaning if no origin is explicitly specified when reading the content or checking the existence of an SAP object type.
- **LOCAL**: Can be obtained via `xco_cp_sap_object_type=>origin->local` and will read a given SAP object type from the local system. By default, every read of the content as well as existence checks of an SAP object type will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_NONT_CACHEABLE_ORIGIN`.

Next, when reading an SAP object type or checking its existence, it can explicitly be specified which version of the SAP object type should be considered. The following versions are supported:

- **ACTIVE**: Can be obtained via `xco_cp_sap_object_type=>version->active`. The active version is used by default, meaning if no version is explicitly specified when reading the content of an SAP object type or checking its existence.
- **INACTIVE**: Can be obtained via `xco_cp_sap_object_type=>version->inactive`.

Following the general style of the XCO Read APIs, each part of an SAP object type can be read out separately:

### Header

Reading out the header of an SAP object type can be done like this:

#### Sample Code

```
DATA(lo_sap_object_type) = xco_cp_abap_repository=>object->ront-
>for( '$SAP_OBJECT_TYPE_NAME$' ).

" Read out the complete header of the active version of the SAP object type
" using the default origin:
DATA(ls_header) = lo_sap_object_type->content( )->get( ).

" Read out only the short description of the inactive version of the SAP
" object type using an explicitly obtained local origin (without caching).
DATA(lo_local_origin) = xco_cp_sap_object_type=>origin->local( ).

DATA(lv_short_description) = lo_sap_object_type-
>content( xco_cp_sap_object_type=>version->inactive
 )->get_short_description( lo_local_origin ).
```

## Generation APIs

SAP object types are integrated with the XCO Generation APIs with the following operations being offered currently for SAP object types:

- **PUT**: Create or update an SAP object type based on an entire content specification so that after running the operation successfully, the SAP object type exists and its content is exactly as specified

- **PATCH**: Change an existing SAP object type by applying the changes which have been provided in a change specification
- **DELETE**: Delete an SAP object type (if it exists)

## PUT Operations

A form specification is used to specify the entire content of an SAP object type which is the basis for running a PUT operation. Upon running the PUT operation, depending on whether the SAP object type already exists or not, it will either be newly created or updated. This is so that after the successful completion of the PUT operation, the SAP object type exists with the content resembling exactly what was specified in the provided form specification. Consider the following code sample which illustrates how to run a PUT operation for an SAP object type:

### Sample Code

```
DATA(lo_put_operation) = xco_cp_generation=>environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-ront->add_object( '$SAP_OBJECT_TYPE_NAME$'
    )->set_package( '$PACKAGE_NAME$'
    )->create_form_specification( ).

lo_form_specification->set_short_description( 'My short description'
    )->set_type_category( xco_cp_sap_object_type=>type_category->business_object
    )->set_name( '$sapObjectTypeName$'
    )->set_object_type_code( '00000' ).

lo_put_operation->execute( ).
```

## PATCH Operations

A change specification is used to specify the changes that should be applied to an existing SAP object type and is the basis for running a PATCH operation for an SAP object type. Upon running a PATCH operation, the changes will be applied to the SAP object type. The following code example illustrates selected changes and how they can be applied via a PATCH operation:

### Sample Code

```
DATA(lo_patch_operation) = xco_cp_generation=>environment->dev_system( '$TRANSPORT_REQUEST$'
    )->create_patch_operation( ).

DATA(lo_change_specification) = lo_patch_operation->for-ront->add_object( '$SAP_OBJECT_TYPE_NAME$'
    )->create_change_specification( ).

" Update the short description.
lo_change_specification->for-update->set_short_description( 'New short
description' ).

lo_patch_operation->execute( ).
```

## DELETE Operations

SAP object types can be deleted via DELETE operations. As per the standard semantics of DELETE operations, if the SAP object type for which a DELETE operation is being run doesn't exist, no action will be performed (and

no error will be raised). If the SAP object type exists, it will be deleted. The code sample below illustrates how to run a DELETE operation for an SAP object type:

#### ↳ Sample Code

```
DATA(lo_delete_operation) = xco_cp_generation->environment->dev_system( '$TRANSPORT_REQUEST$' )->for_ront->create_delete_operation( ).  
lo_delete_operation->add_object( '$SAP_OBJECT_TYPE_NAME$' ).  
lo_delete_operation->execute( ).
```

### 4.2.11.16.3.8.7 Service Bindings

Find out how to create service bindings.

Service bindings, also called objects of type SRVB, are supported by the following API families within the [XCO Library](#):

- Query APIs
- Read APIs
- Generation APIs

In addition to the standard API families listed above, dedicated APIs are offered for the following aspects:

- Publishing and unpublishing local service endpoints

### Query APIs

Obtaining an object handle for a service binding with a given name can be accomplished via

#### ↳ Sample Code

```
DATA(lo_service_binding) = xco_cp_abap_repository->object->srvb->for( '$SERVICE_BINDING_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the service binding in the ABAP Repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible service bindings in the ABAP Repository can be accomplished via

#### ↳ Sample Code

```
DATA(lt_service_bindings) = xco_cp_abap_repository->objects->srvb->all->in( xco_cp_abap=>repository )->get( ).  
LOOP AT lt_service_bindings ASSIGNING FIELD-SYMBOL(<fs_service_binding>).  
    " Process the found service binding.  
ENDLOOP.
```

Instead of searching the entire ABAP Repository for service bindings, it's also possible to only search for service bindings in a dedicated subset of the ABAP Repository, such as in a single package:

#### ↳ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).  
DATA(lt_service_bindings) = xco_cp_abap_repository->objects->srvb->all-  
>in( lo_package )->get( ).  
  
LOOP AT lt_service_bindings ASSIGNING FIELD-SYMBOL(<fs_service_binding>).  
    " Process the found service binding.  
ENDLOOP.
```

Another way to refine a search for service bindings is to use filters to specify exactly which service bindings shall be matched, for example:

#### ↳ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(  
    xco_cp_abap_sql->constraint->contains_pattern( '/ABC/%' )  
).  
  
" LT_SERVICE_BINDINGS will be a list of all visible service bindings in the  
ABAP  
" Repository whose name starts with /ABC/.  
DATA(lt_service_bindings) = xco_cp_abap_repository->objects->srvb-  
>where( VALUE #(  
    ( lo_object_name_filter )  
) ->in( xco_cp_abap->repository )->get( ).  
  
LOOP AT lt_service_bindings ASSIGNING FIELD-SYMBOL(<fs_service_binding>).  
    " Process the found service binding.  
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of a service binding is possible via its object handle `IF_XCO_SERVICE_BINDING`. The Read APIs for service bindings are based on the origin infrastructure with the following origins being currently offered for service bindings:

- **DEFAULT**: Can be obtained via `XCO_CP_SERVICE_BINDING=>ORIGIN=>DEFAULT` and will read a given service binding exactly once per ABAP session from the local system with all subsequent reads of the service binding being done from a dedicated buffer. As the name suggests, this origin is used by default, so if no origin is explicitly specified when reading the content or checking the existence of a service binding.
- **LOCAL**: Can be obtained via `XCO_CP_SERVICE_BINDING=>ORIGIN=>LOCAL` and will read a given service binding from the local system. By default, every read of the content and check of the existence of a service binding will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_SRVB_CACHEABLE_ORIGIN`.

Furthermore, when reading a service binding or checking its existence, it can explicitly be specified which version of the service binding to consider. The following versions are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_SERVICE_BINDING=>VERSION=>ACTIVE`. The active version is used by default, so if no version is explicitly specified when reading the content or checking the existence of a service binding.
- **INACTIVE** : Can be obtained via `XCO_CP_SERVICE_BINDING=>VERSION=>INACTIVE`.

Following the general style of the XCO Read APIs, each part of a service binding can be read out separately:

## Header

Reading out the header of a service binding can be done like this:

### Sample Code

```
DATA(lo_service_binding) = xco_cp_abap_repository=>object->srvb-
>for( '$SERVICE_BINDING_NAME$' ).

" Read out the complete header of the active version of the service binding
" using the default origin:
DATA(ls_header) = lo_service_binding->content( )->get( ).

" Read out only the short description of the inactive version of the service
" binding using an explicitly obtained local origin (without caching).
DATA(lo_local_origin) = xco_cp_service_binding=>origin->local( ).

DATA(lv_short_description) = lo_service_binding-
>content( xco_cp_service_binding=>version->inactive
 )->get_short_description( lo_local_origin ).
```

## Services

Obtaining a list of the services of a service binding can be done like this:

### Sample Code

```
DATA(lo_service_binding) = xco_cp_abap_repository=>object->srvb-
>for( '$SERVICE_BINDING_NAME$' ).

" Get a list of all the services of the active version of the service binding
using
" the default origin:
DATA(lt_services) = lo_service_binding->services->all->get( ).

LOOP AT lt_services ASSIGNING FIELD-SYMBOL(<fs_service>).
    " Process the service.
ENDLOOP.
```

## Service Versions

Obtaining a list of the service versions of a service and reading out an individual service version can be done like this:

### Sample Code

```
" Note that in this example we would be reading the content of a service
" with a binding type for which services can be added freely (e.g. an OData v4
" service binding).
```

```

DATA(lo_service) = xco_cp_abap_repository->object->srvb-
>for( '$SERVICE_BINDING_NAME$'
  )->service( '$SERVICE_NAME$' ).

" Get a list of all the service versions of the service for the active version
" of the service binding using the default origin.
DATA(lt_service_versions) = lo_service->versions->all->get( ).

LOOP AT lt_service_versions ASSIGNING FIELD-SYMBOL(<fs_service_version>).
  " Process the service version.
ENDLOOP.

" Obtain the handle for an individual service version.
DATA(lo_service_version) = lo_service->version( 0001 ).

" Read out the complete content of the service version for the active version
of the
" service binding using the default origin.
DATA(ls_service_version) = lo_service_version->content( )->get( ).

" Read out only the service definition of the service version for the
inactive version
" of the service binding using an explicitly obtained local origin (with
caching enabled).
DATA(lo_local_origin) = xco_cp_service_binding->origin->local( ).
lo_local_origin->if_xco_srvb_cacheable_origin~enable_caching( ).

DATA(lo_service_definition) = lo_service_version-
>content( xco_cp_service_binding=>version->inactive
)->get_service_definition( lo_local_origin ).
```

## Generation APIs

Service bindings are integrated with the XCO Generation APIs with the following operations being currently offered for service bindings:

- [PUT](#): Create or update a service binding based on a complete specification of its content, so that after successfully running the operation, the service binding exists and its content is exactly as specified.
- [DELETE](#): Delete a service binding (if it exists).

### PUT Operations

A form specification is used to specify the entire content of a service binding, which is the foundation for running a `PUT` operation. Upon execution of the `PUT` operation, depending on whether the service binding already exists or not, it will either be newly created or updated. This is so that after the successful completion of the `PUT` operation, the service binding exists with the content resembling exactly what was specified in the provided form specification. Consider the following code sample which illustrates how to run a `PUT` operation for a service binding:

#### Sample Code

```

DATA(lo_put_operation) = xco_cp_generation->environment-
>dev_system( '$TRANSPORT_REQUEST$'
  )->create_put_operation( ).

DATA(lo_form_specification) = lo_put_operation->for-srvb-
>add_object( '$SERVICE_BINDING_NAME$'
  )->set_package( '$PACKAGE_NAME$'
```

```

) -> create_form_specification( ).

lo_form_specification -> set_short_description( 'My short description'
) -> set_binding_type( xco_cp_service_binding -> binding_type -> odata_v4_ui ).

DATA( lo_service ) = lo_form_specification -> add_service( '$SERVICE_NAME$' ).

DATA( lo_service_version_0001 ) = lo_service -> add_version( 0001 ).

lo_service_version_0001 -
> set_service_definition( '$SERVICE_DEFINITION_NAME$' ).

lo_put_operation -> execute( ).

```

## DELETE Operations

Service bindings can be deleted via **DELETE** operations. As per the standard semantics of **DELETE** operations, if the service binding for which a **DELETE** operation is run doesn't exist, no action will be performed (and no error will be raised). If the service binding does exist, it will be deleted. The code sample below illustrates how to run a **DELETE** operation for a service binding:

### ↔ Sample Code

```

DATA( lo_delete_operation ) = xco_cp_generation -> environment-
> dev_system( '$TRANSPORT_REQUEST'
) -> for-srvb -> create_delete_operation( ).

lo_delete_operation -> add_object( '$SERVICE_BINDING_NAME$' ).

lo_delete_operation -> execute( ).

```

## Publishing and Unpublishing Local Service Endpoints

For service bindings with binding types OData V2 - UI/Web API or OData V4 - UI/Web API, it's possible to publish (or unpublish) a local service endpoint for the service binding. The exact granularity of a local service endpoint depends on whether the binding type belongs to the OData V2 or V4 binding type family:

### Local Service Endpoints and OData V2 Service Bindings

For service bindings with binding type OData V2 - UI or OData V2 - Web API, a local service endpoint can be published (or unpublished) for each (service) version defined in the service binding. To programmatically trigger the publish (or unpublish) of a local service endpoint, the [XCO Library](#) provides designated operations which are available via `XCO_CP_SERVICE_BINDING->LOCAL_SERVICE_ENDPOINT->ODATA_V2->OPERATION`, namely:

- **PUBLISH**: Triggers a publish of the local service endpoint of the provided service version
- **UNPUBLISH**: Triggers an unpublish of the local service endpoint of the provided service version

Moreover, it's possible to check whether the local service endpoint of a given service version is currently published via `XCO_CP_SERVICE_BINDING->LOCAL_SERVICE_ENDPOINT->ODATA_V2->IS_PUBLISHED`. The following code illustrates the use of the operations as well as of the `IS_PUBLISHED` method:

## ↔ Sample Code

```
DATA(lo_service_version) = xco_cp_abap_repository=>object->srvb-
>for( '$SERVICE_BINDING_NAME$'
 )->service(
 )->version( 0001 ).

" First we check whether the local service endpoint for LO_SERVICE_VERSION is
" currently published.
DATA(lv_is_published) = xco_cp_service_binding=>local_service_endpoint-
>odata_v2->is_published( lo_service_version ).

" Depending on whether the local service endpoint is currently published we
either
" trigger an unpublish or a publish of the local service endpoint.
DATA lo_operation TYPE REF TO if_xco_srvb_operation.

IF lv_is_published EQ abap_true.
  lo_operation = xco_cp_service_binding=>local_service_endpoint->odata_v2-
>operation->unpublish( lo_service_version ).
ELSE.
  lo_operation = xco_cp_service_binding=>local_service_endpoint->odata_v2-
>operation->publish( lo_service_version ).
ENDIF.

" Note that both a publish as well as an unpublish operation are concrete
" realizations of an IF_XCO_SRVB_OPERATION and can thus be treated uniformly.
" Regardless of the concrete realization of the operation, its execution can
" always be triggered via the method EXECUTE.
lo_operation->execute( ).
```

## Local Service Endpoints for OData V4 Service Bindings

For service bindings with binding type OData V4 - UI or OData V4 - Web API, a local service endpoint can only be published (or unpublished) for the entire service binding. To programmatically trigger the publish (or unpublish) of a local service endpoint, the [XCO Library](#) provides designated operations which are available via XCO\_CP\_SERVICE\_BINDING=>LOCAL\_SERVICE\_ENDPOINT->ODATA\_V4->OPERATION, namely:

- **PUBLISH**: Triggers a publish of the local service endpoint of the provided service binding
- **UNPUBLISH**: Triggers an unpublish of the local service endpoint of the provided service binding

Moreover, it's possible to check whether the local service endpoint of a given service binding is currently published via XCO\_CP\_SERVICE\_BINDING=>LOCAL\_SERVICE\_ENDPOINT->ODATA\_V4->IS\_PUBLISHED. The following code illustrates the use of the operations as well as of the IS\_PUBLISHED method:

## ↔ Sample Code

```
DATA(lo_service_binding) = xco_cp_abap_repository=>object->srvb-
>for( '$SERVICE_BINDING_NAME$' ).

" First we check whether the local service endpoint for LO_SERVICE_BINDING is
" currently published.
DATA(lv_is_published) = xco_cp_service_binding=>local_service_endpoint-
>odata_v4->is_published( lo_service_binding ).

" Depending on whether the local service endpoint is currently published we
either
" trigger an unpublish or a publish of the local service endpoint.
DATA lo_operation TYPE REF TO if_xco_srvb_operation.

IF lv_is_published EQ abap_true.
  lo_operation = xco_cp_service_binding=>local_service_endpoint->odata_v4-
>operation->unpublish( lo_service_binding ).
```

```

ELSE.
  lo_operation = xco_cp_service_binding=>local_service_endpoint->odata_v4-
>operation->publish( lo_service_binding ).
ENDIF.

" Note that both a publish as well as an unpublish operation are concrete
" realizations of an IF_XCO_SRVBL_OPERATION and can thus be treated uniformly.
" Regardless of the concrete realization of the operation, its execution can
" always be triggered via the method EXECUTE.
lo_operation->execute( ).
```

## 4.2.11.16.3.8.8 Service Consumption Models

Find out how to create service consumption models.

Service consumption models, also called objects of type SRVC, are supported by the following API families within the [XCO Library](#):

- Query APIs
- Read APIs

### Query APIs

Obtaining an object handle for a service consumption model with a given name can be accomplished via

#### ↔ Sample Code

```
DATA(lo_service_consumption_model) = xco_cp_abap_repository=>object->srvc-
>for( '$SRVC_NAME$' ).
```

The obtained object handle can then be used to, for example, check the existence of the service consumption model in the ABAP Repository (via method `IF_XCO_AR_OBJECT~EXISTS`) or read out its content via the Read APIs (see below). Obtaining a list of all visible service consumption models in the ABAP Repository can be accomplished via

#### ↔ Sample Code

```
DATA(lt_service_consumption_models) = xco_cp_abap_repository=>objects->srvc-
>all->in( xco_cp_abap=>repository )->get( ).

LOOP AT lt_service_consumption_models ASSIGNING FIELD-
SYMBOL(<fs_service_consumption_model>).
  " Process the found service consumption model.
ENDLOOP.
```

Instead of searching the entire ABAP Repository for service consumption models, it's also possible to only search for service consumption models in a dedicated subset of the ABAP Repository, such as in a single package:

### ↔ Sample Code

```
DATA(lo_package) = xco_cp_abap_repository->package->for( '$PACKAGE_NAME$' ).  
DATA(lt_service_consumption_models) = xco_cp_abap_repository->objects->srvc->all->in( lo_package )->get( ).  
  
LOOP AT lt_service_consumption_models ASSIGNING FIELD-SYMBOL(<fs_service_consumption_model>).  
    " Process the found service consumption model.  
ENDLOOP.
```

Another way to refine a search for service consumption models is to use filters to specify exactly which service consumption models shall be matched, for example:

### ↔ Sample Code

```
DATA(lo_object_name_filter) = xco_cp_abap_repository->object_name->get_filter(  
    xco_cp_abap_sql->constraint->contains_pattern( '/ABC/%' )  
).  
  
" LT_SERVICE_CONSUMPTION_MODELS will be a list of all visible service  
consumption  
" models in the ABAP Repository whose name starts with /ABC/.  
DATA(lt_service_consumption_models) = xco_cp_abap_repository->objects->srvc->where( VALUE #(  
    ( lo_object_name_filter )  
) )->in( xco_cp_abap->repository )->get( ).  
  
LOOP AT lt_service_consumption_models ASSIGNING FIELD-SYMBOL(<fs_service_consumption_model>).  
    " Process the found service consumption model.  
ENDLOOP.
```

## Read APIs

Reading out the content and checking the existence of a service consumption model is possible via its object handle `IF_XCO_SERVICE_CONSUMPTION_MDL`. The Read APIs for service consumption models are based on the origin infrastructure with the following origins being currently offered for service consumption models:

- **DEFAULT**: Can be obtained via `XCO_CP_SERVICE_CONS_MODEL=>ORIGIN=DEFAULT` and will read a given service consumption model exactly once per ABAP session from the local system, with all subsequent reads of the service consumption model being done from a dedicated buffer. As the name suggests, this origin is used by default, so if no origin is explicitly specified when reading the content or checking the existence of a service consumption model.
- **LOCAL**: Can be obtained via `XCO_CP_SERVICE_CONS_MODEL=>ORIGIN=LOCAL` and will read a given service consumption model from the local system. By default, every read of a service consumption model will read directly from the database, but the caching behavior can be configured via the methods of `IF_XCO_SRVC_CACHEABLE_ORIGIN`.

Furthermore, when reading a service consumption model or checking its existence, it can explicitly be specified which version of the service consumption model shall be considered. The following versions are supported:

- **ACTIVE**: Can be obtained via `XCO_CP_SERVICE_CONS_MODEL=>VERSION=ACTIVE`. The active version is used by default, so if no version is explicitly specified when reading the content of a service consumption model or checking its existence.
- **INACTIVE**: Can be obtained via `XCO_CP_SERVICE_CONS_MODEL=>VERSION=INACTIVE`.

Following the general style of the XCO Read APIs, each part of a service consumption model can be read out separately:

## Header

Reading out the header of a service consumption model can be done like this:

### ↳ Sample Code

```
DATA(lo_service_consumption_model) = xco_cp_abap_repository=>object->srvc->for( '$SRVC_NAME$' ).  
  
" Read out the complete header of the active version of the service  
consumption model  
" using the default origin:  
DATA(ls_header) = lo_service_consumption_model->content( )->get( ).  
  
" Read out only the short description of the inactive version of the service  
" consumption model using an explicitly obtained local origin (without  
caching).  
DATA(lo_local_origin) = xco_cp_service_cons_model=>origin->local( ).  
  
DATA(lv_short_description) = lo_service_consumption_model->content( xco_cp_service_cons_model=>version->inactive  
)->get_short_description( lo_local_origin ).
```

## Service Entity Sets

For service consumption models with consumption type ODATA (the corresponding XCO enumeration constant is obtainable via `XCO_CP_SERVICE_CONS_MODEL=>CONSUMPTION_TYPE=ODATA`), it's possible to read the list of service entity sets of the service consumption model as well as the details of a given service entity set:

### ↳ Sample Code

```
DATA(lo_service_consumption_model) = xco_cp_abap_repository=>object->srvc->for( '$SRVC_NAME$' ).  
  
" Get a list of all the service entity sets of the active version of the  
service  
" consumption model using the default origin:  
DATA(lt_service_entity_sets) = lo_service_consumption_model->service_entity_sets->all->get( ).  
  
LOOP AT lt_service_entity_sets ASSIGNING FIELD-  
SYMBOL(<fs_service_entity_set>).  
    " Process the service entity sets.  
ENDLOOP.  
  
" Obtain the handle for an individual service entity set.  
DATA(lo_service_entity_set) = lo_service_consumption_model->service_entity_set( '$$SERVICE_ENTITY_SET$' ).  
  
" Read out the complete content of the service entity set for the active  
version of the service
```

```

" consumption model using the default origin.
DATA(ls_service_entity_set) = lo_service_entity_set->content( )->get( ).

" Read out only the data definition of the service entity set for the
inactive version of the service
" consumption model using an explicitly obtained local origin (with caching
enabled).
DATA(lo_local_origin) = xco_cp_service_cons_model->origin->local( ).
lo_local_origin->if_xco_srvc_cacheable_origin~enable_caching( ).

DATA(lo_data_definition) = lo_service_entity_set-
>content( xco_cp_service_cons_model->version->inactive
)->get_data_definition( lo_local_origin ).
```

## Service Operations

For service consumption models with consumption type WEB\_SERVICE (the corresponding XCO enumeration constant is obtainable via XCO\_CP\_SERVICE\_CONS\_MODEL=>CONSUMPTION\_TYPE>WEB\_SERVICE), it's possible to read the list of service operations of the service consumption model:

### ↔ Sample Code

```

DATA(lo_service_consumption_model) = xco_cp_abap_repository->object->srvc-
>for( '$SRVC_NAME$' ).

" Get a list of all the service operations of the active version of the
service
" consumption model using the default origin:
DATA(lt_service_operations) = lo_service_consumption_model-
>service_operations->all->get( ).

LOOP AT lt_service_operations ASSIGNING FIELD-SYMBOL(<fs_service_operation>).
  " Process the service operations.
ENDLOOP.
```

## Remote Functions

For service consumption models with consumption type RFC (the corresponding XCO enumeration constant is obtainable via XCO\_CP\_SERVICE\_CONS\_MODEL=>CONSUMPTION\_TYPE>RFC), it's possible to read the list of remote functions of the service consumption model:

### ↔ Sample Code

```

DATA(lo_service_consumption_model) = xco_cp_abap_repository->object->srvc-
>for( '$SRVC_NAME$' ).

" Get a list of all the remote functions of the active version of the service
" consumption model using the default origin:
DATA(lt_remote_functions) = lo_service_consumption_model->remote_functions-
>all->get( ).

LOOP AT lt_remote_functions ASSIGNING FIELD-SYMBOL(<fs_remote_function>).
  " Process the remote functions.
ENDLOOP.
```

## 4.2.11.16.3.9 Miscellaneous Code Samples

### 4.2.11.16.3.9.1 Getting the Text Table of a Database Table

The XCO standard abstraction for database tables (`IF_XCO_DATABASE_TABLE`) provides a convenient way to determine the text table of a given database table (if it has one):

#### ↔ Sample Code

```
DATA(lo_database_table) = xco_cp_abap_dictionary->database_table( 'ZMY_DBT' ).  
  
" From a database table handle (in this case LO_DATABASE_TABLE) we can obtain  
" the handle for its text table.  
DATA(lo_text_table) = lo_database_table->text_table( ).  
  
" Using this text table handle, we can first check if a text table exists at  
all  
" (for the previously fixed database table) and if so, obtain information  
about  
" it.  
DATA(lv_text_table_exists) = lo_text_table->exists( ).  
  
IF lv_text_table_exists EQ abap_true.  
" The name of the text table that belongs to database table ZMY_DBT.  
DATA(lv_text_table_name) = lo_text_table->get( )->name.  
  
" We can also use the text table handle to get the name of the check field  
of the  
" database table/text table relationship.  
DATA(lv_check_field_name) = lo_text_table->get_check_field( )->name.  
ENDIF.
```

### 4.2.11.16.3.9.2 Retrieving a List of All Classes Implementing an Interface

The standard abstraction provided by the XCO library for ABAP interfaces (`IF_XCO_AO_INTERFACE`) can be used to retrieve a list of all implementations of a given interface:

#### ↔ Sample Code

```
DATA(lo_interface) = xco_cp_abap->interface( 'ZMY_INTERFACE' ).  
  
" From an interface handle (in this case LO_INTERFACE) we can easily obtain  
" access to its implementations via the IMPLEMENTATIONS data member. Obtaining  
" a list of all implementations for an interface can be accomplished like  
this:  
DATA(lt_implementations) = lo_interface->implementations->all->get( ).  
  
" Once the previous statement has been executed, LT_IMPLEMENTATIONS will be a  
" table of IF_XCO_AO_CLASS objects which can then subsequently be used to e.g.  
" read the content of each class. Note that only those implementations will be  
" considered which are either C1-released or part of a customer software  
component.  
" Additionally, if only the names of the classes implementing the interface
```

```
" shall be obtained the following statement can be used:  
DATA(lt_implementation_names) = lo_interface->implementations->all->get_names( ).
```

### 4.2.11.16.3.9.3 Retrieving a List of all Subclasses of a Given Class

Find out how to retrieve a list of all subclasses of a given class.

The standard abstraction provided by the XCO library for ABAP classes (`IF_XCO_AO_CLASS`) can be used to retrieve a list of all subclasses (including both direct and indirect subclasses) of a given class:

#### ↔ Sample Code

```
DATA(lo_class) = xco_cp_abap=>class( 'ZMY_CLASS' ).  
  
" From a class handle (in this case LO_CLASS) we can easily obtain access to  
" its subclasses via the SUBCLASSES data member. Accessing all subclasses  
" via SUBCLASSES->ALL will resolve the subclasses of the class recursively  
" along the inheritance tree, i.e. all subclasses including subclasses of  
" subclasses and so forth will be included.  
DATA(lt_subclasses) = lo_class->subclasses->all->get( ).  
  
" Once the previous statement has been executed, LT_SUBCLASSES will be a  
" table of IF_XCO_AO_CLASS objects which can then subsequently be used to  
" e.g. read the content of each class. Furthermore, note that only those  
" subclasses will be considered which are either C1-released or part of a  
" customer software component. If only the names of the subclasses shall be  
" obtained, the following variation can be used:  
DATA(lt_subclass_names) = lo_class->subclasses->all->get_names( ).
```

### 4.2.11.16.4 Standard Library

The XCO standard library provides lightweight abstractions to simplify everyday programming tasks and defines central abstractions, most notably `IF_XCO_NEWS` and `IF_XCO_TEXT`, that provide a common language to foster synergies between different XCO modules.

## Building blocks

There are several basic building blocks defined within the XCO standard library that are used throughout the modules of the whole XCO library and may even be integrated into custom coding.

### Exception philosophy

Unless a caller can be reasonably expected to recover from an error situation is always handled through `NO_CHECK` exceptions that are derived from `CX_XCO_RUNTIME_EXCEPTION`.

The class CX\_XCO\_RUNTIME\_EXCEPTION is an abstract class that defines aspects that are common to all kinds of unexpected error situations. Most importantly, it provides access to the messages of the exceptional situation via IF\_XCO\_NEWS.

On top of that, every time a CX\_XCO\_RUNTIME\_EXCEPTION is raised the active call stack is captured when the exception object is constructed. By having exception classes inherit from CX\_XCO\_RUNTIME\_EXCEPTION stack traces can also be easily enabled for self-defined exceptions.

When such an exception is logged via the XCO BAL module the stack trace will be included along with the message of the exception (assuming the XCO\_CP\_BAL=>EXCEPTION\_ADDITION->STACK\_TRACE exception addition has been enabled).

## News

The interface IF\_XCO\_NEWS provides a central abstraction that is applicable to any object that can naturally act as a source of messages. A message is defined as an object of type IF\_XCO\_MESSAGE which in turn is characterized by a SYMSG value.

As the interface IF\_XCO\_NEWS can be flexibly mixed in with existing class hierarchies it provides a common denominator that enables communication between independent modules both within the XCO library as well as between the XCO library and custom coding.

A good example for the synergies created by the IF\_XCO\_NEWS abstraction is the interplay of the XCO BAL module with the XCO ADT standard library functionality which is described below. Even though the XCO ADT functionality has no knowledge and is completely independent of the XCO BAL module writing all the messages contained in a log to the console is as simple as

### ↔ Sample Code

```
out->write( lo_log->messages->all )
```

Notable implementations of IF\_XCO\_NEWS are:

- IF\_XCO\_RAP\_BEHAVIOR\_MESSAGE. Obtainable via XCO\_CP\_RAP=>BEHAVIOR MESSAGE( ... ), an IF\_XCO\_RAP\_BEHAVIOR\_MESSAGE encapsulates an IF\_ABAP\_BEHV\_MESSAGE and allows it to be used anywhere where an IF\_XCO\_NEWS object can be used.

## Text

Closely related to the IF\_XCO\_NEWS abstraction is the IF\_XCO\_TEXT interface. The main difference is that it represents an object than can naturally act as a source of strings (as compared to an object that can naturally act as a source of messages as is the case with IF\_XCO\_NEWS).

Like IF\_XCO\_NEWS it provides a common ground for many different XCO modules. It is furthermore the basis of the IF\_XCO\_PRINTABLE interface which allows to define a print version for any object that is used when the object is written to the console via the XCO ADT functionality (see [ADT \[page 1792\]](#)).

## Related Information

[ADT \[page 1792\]](#)

[JSON \[page 1800\]](#)

[Regular Expression \[page 1802\]](#)

[String \[page 1803\]](#)

[UUID \[page 1809\]](#)

## 4.2.11.16.4.1 ADT

By having a class implement the interface IF\_OO\_ADT\_CLASSRUN it is possible to easily write programs that can be run directly within ADT using the console as output.

The XCO library provides an abstract implementation of this interface via CL\_XCO\_CP\_ADT\_SIMPLE\_CLASSRUN that exposes all the original functionality of IF\_OO\_ADT\_CLASSRUN but adds several additional functionalities that enable a natural integration of XCO library abstractions with ADT classruns.

The following sample class illustrates how the OUT object provided by CL\_XCO\_CP\_ADT\_SIMPL\_CLASSRUN can be used to flexibly write complex data structures or even object references to the console. Using the interface IF\_XCO\_PRINTABLE any object can define a text that is used when the object is written to the console.

### ↔ Sample Code

```
CLASS zcl_xco_doc_cp_std_adt DEFINITION PUBLIC FINAL
  INHERITING FROM cl_xco_cp_adt_simple_classrun CREATE PUBLIC.
  PROTECTED SECTION.
  METHODS:
    main REDEFINITION.
  ENDCLASS.

CLASS zcl_xco_doc_cp_std_adt IMPLEMENTATION.
METHOD main.
TYPES:
  BEGIN OF ts_address,
    street      TYPE string,
    house_number TYPE n LENGTH 2,
    postal_code  TYPE n LENGTH 5,
    city        TYPE string,
  END OF ts_address,
  BEGIN OF ts_member,
    first_name  TYPE string,
    last_name   TYPE string,
    address     TYPE ts_address,
  END OF ts_member,
  tt_member TYPE STANDARD TABLE OF ts_member WITH EMPTY KEY.
  " Print a JSON-style serialization (works for arbitrarily deep data
structures)
DATA(lt_members) = VALUE tt_member(
  ( first_name = 'John' last_name = 'Doe' address = VALUE #( street =
'Main street' house_number = 12 postal_code = 12345 city = 'City' ) )
).
  " Will print: [{FIRST_NAME: John, LAST_NAME: Doe, ADDRESS: {STREET: Main
street, HOUSE_NUMBER: 12, POSTAL_CODE: 12345, CITY: City}}]
  out->write( lt_members ).
  " IF_XCO_PRINTABLE example: Will print "WARNING"
  out->write( xco_cp_message=>type->warning ).
  " IF_XCO_PRINTABLE example: Will print "Field MY_FIELD of View entity
MY_VIEW_ENTITY"
  DATA(lo_field) = xco_cp_cds=>view_entity( 'MY_VIEW_ENTITY' )-
>field( 'MY_FIELD' ).
  out->write( lo_field ).
ENDMETHOD.
```

```
ENDCLASS.
```

On top of the rich console writing functionalities, any NO\_CHECK exception that is not caught by the MAIN method of a class implementing CL\_XCO\_CP\_ADT\_SIMPLE\_CLASSRUN will be caught and written to the console with maximum information. This includes the raise position, messages and even the stack trace if the exception is carrying one (cf. explanation of CX\_XCO\_RUNTIME\_EXCEPTION above).

## 4.2.11.16.4.2 Business Application Log

The XCO BAL module provides APIs and abstractions that allow a smooth integration of logging into application logic based on the standard Business Application Log (BAL) and provide maximum synergy with the facilities offered by the XCO library in general.

### Terminology

Within the BAL module of the XCO Library, the following terminology is used:

- **Log object:** The design time ABAP Repository object (type APLO) defining an application log object together with its subobjects.
- **Log:** A single log instance for a given log object and subobject with an optional external ID into which messages can be written and read from. Represented by an object of type IF\_XCO\_CP\_BAL\_LOG and uniquely identified by a log handle.
- **Log handle:** A unique identifier for a log (UUID in CHAR22 format) associated with a log (available through attribute LOG\_HANDLE of interface IF\_XCO\_CP\_BAL\_LOG).

### Persistence

The first and one of the central abstractions within the XCO BAL module is that of the persistence given by the interface IF\_XCO\_CP\_BAL\_PERSISTENCE. A persistence decides where logs are created (resp. loaded from) and where the messages and exceptions are written to (resp. read from).

The following two flavors are offered:

- Memory: Logs are created only in memory and messages as well as exceptions are not written to the database
- Database: Logs are created and loaded from the database and messages and exceptions are always saved to the database. Whenever a message or exception is added to the log a COMMIT WORK is performed on a secondary database connection

## Searching logs

In a style like that of the XCO ABAP Repository Query APIs it is possible to easily locate existing logs based on filters for standard log attributes, like the object, subobject or external ID of a log:

### ↳ Sample Code

```
DATA(lo_external_id_filter) = xco_cp_bal->log_filter->external_id(  
    xco_cp_abap_sql->constraint->equal( '%INVOICE%' )  
).  
DATA(lt_logs) = xco_cp_bal->for->database( )->logs->where( VALUE #(  
    ( lo_external_id_filter )  
) )->get( ).
```

In this example, all logs in the database whose external ID contains the fragment INVOICE will be located.

## Creating and loading logs

Once the desired persistence has been selected via XCO\_CP\_BAL=>FOR a log can be created like

### ↳ Sample Code

```
DATA(lo_log) = xco_cp_bal->for->database( )->log->create(  
    iv_object      = 'MY_LOG_OBJECT'  
    iv_subobject   = 'MY_LOG_SUBOBJECT'  
    iv_external_id = 'MY_EXTERNAL_ID'  
).
```

If a log shall not be created newly but already exists in the database it can be loaded like

### ↳ Sample Code

```
DATA(lo_log) = xco_cp_bal->for->database( )->log->load( 'MY_LOG_HANDLE' ).
```

for a given log handle.

## Profiles

Associated with each IF\_XCO\_CP\_BAL\_LOG object is a profile (IF\_XCO\_CP\_BAL\_PROFILE) that influences how messages and exceptions are added to the log. For messages, a profile provides a default value for the level of detail that is used when messages are added to the log and no explicit level of detail is supplied.

For exceptions, the XCO BAL module offers a rich set of extra information that can be added along with the exception message to achieve maximum information that can be used for detailed error analysis.

The standard profile (XCO\_CP\_BAL=>PROFILE->STANDARD) defines a default level of detail of 4 and includes all possible exception additions as well as an automatic recursive descent for exceptions, i.e. all the previous exceptions of an exception are added to the log as well.

## Adding and reading messages and exceptions

Messages can be added to a log in number of ways. They can be added directly via an explicit SYMSG value like

### ↳ Sample Code

```
" Logging a plain SYMSG using the level of detail given by the underlying
" profile.
DATA(ls_symmsg) = VALUE symsg(
    msgty = 'W'
    msgid = 'MSG_CLASS'
    msgno = '001'
    msgvl = 'Item 1'
).
lo_log->add_message( ls_symmsg ).
" Logging the same message but this time with an explicitly specified
" level of detail.
lo_log->add_message(
    is_symsg      = ls_symmsg
    io_level_of_detail = xco_cp_bal=>level_of_detail->nine
).
```

or by resorting to the standard abstractions IF\_XCO\_NEWS and IF\_XCO\_TEXT from the XCO standard library:

### ↳ Sample Code

```
" Support for IF_XCO_NEWS and IF_XCO_TEXT provides a high degree of
integration with other
parts of the XCO library.
MESSAGE w001(msg_class) WITH |Item 1| INTO DATA(lv_message).
lo_log->add_news( xco_cp=>sy ).
DATA(lv_string) = |MyStringValue|.
lo_log->add_text( xco_cp=>string( lv_string ) ).
```

Following the pattern which is also used when reading the content of an object of the ABAP Repository, it is easy to access all the messages contained in a log:

### ↳ Sample Code

```
DATA(lt_messages) = lo_log->messages->all->get( ).
```

### ⓘ Note

Note that in order to be able to read from a log, an authorization is required for authorization object S\_APPL\_LOG for the following authorization field values:

- ALG\_OBJECT The object of the application log that is being read
- ALG\_SUBOBJ The subobject of the application log that is being read
- ACTVT: 03

Adding and getting exceptions is identical to how messages are added to and read from the log: Exceptions are added via the ADD\_EXCEPTION method and read via the EXCEPTIONS attribute of an IF\_XCO\_CP\_BAL\_LOG object.

Note that when an exception is added to a log it will automatically be transformed into a message by the underlying Business Application Log if it provides a T100 message. Only when an exception does not provide a T100 message will it be available via the EXCEPTIONS read attribute of IF\_XCO\_CP\_BAL\_LOG.

## 4.2.11.16.4.3 Call Stack

The XCO call stack module allows to programmatically capture and process ABAP call stacks during execution.

The following example call stack, captured by running a locally created ADT classrun (ZCL\_XCO\_CALL\_STACK) will be used to illustrate the different functionalities.

Line	Positive position	Negative position
ZCL_XCO_CALL_STACK main [method]	1	-17
CL_XCO_ADT_CR_SIMPLE_ACTION execute [method]	2	-16
CL_XCO_DP_ACTION_ABSTRACT if_xco_dp_action~execute [method]	3	-15
CL_XCO_ADT_CALL_CLASSRUN if_xco_rt_adt_call_classrun~dispatch [method]	4	-14
CL_XCO_RT_ADT_CLASSRUN_ABSTR if_oo_adt_classrun~main [method]	5	-13
CL_OO_ADT_RES_CLASSRUN execute_clas [method]	6	-12
CL_OO_ADT_RES_CLASSRUN post [method]	7	-11
CL_ADT_REST_RESOURCE if_rest_handler~handle [method]	8	-10
CL_REST_ROUTER if_rest_handler~handle [method]	9	-9
CL_REST_ROUTER if_rest_handler~handle [method]	10	-8
CL_REST_HTTP_HANDLER if_http_extension~handle_request [method]	11	-7
CL_ADT_WB_RES_APP lif_request_handler~handle_request [method]	12	-6
CL_ADT_WB_RES_APP handle_request [method]	13	-5
CL_ADT_WB_RES_APP if_http_extension~handle_request	14	-4
CL_HTTP_SERVER [system] execute_request [method]	15	-3

Line	Positive position	Negative position
SAPLHTTP_RUNTIME [system] http_dispatch_request [function]	16	-2
SAPMHTTP [system] %_http_start [module (pbo)]	17	-1

In a manner analogous to how characters of a string are indexed when substrings are extracted via `IF_XCO_STRING`, the XCO call stack module associates both a positive and a negative index/position with each call stack entry as illustrated above.

## Capturing the current call stack

The current call stack can be captured by using the API `XCO_CP=>CURRENT->CALL_STACK`. The following two methods are available:

- FULL: Captures the full current call stack, i.e. all available entries up the lowest entry point will be included
- UP\_TO: Captures the current call stack up to the given depth, i.e. only the topmost `IV_DEPTH` call stack entries will be captured

Once captured, a call stack is represented as an immutable object of type `IF_XCO_CP_CALL_STACK`.

## Extracting parts of a call stack

Given a call stack (represented as an object of type `IF_XCO_CP_CALL_STACK`) various ways are provided to extract specific parts of the call stack. As the result of any extraction is again an object of type `IF_XCO_CP_CALL_STACK` it's easy to cascade multiple extractions to form complex and highly customized extractions.

The following kinds of extractions are provided:

- POSITION: Uses a position (either positive or negative) to fix a reference call stack entry
- FIRST\_OCCURRENCE\_OF: Uses the first occurrence of a given call stack line pattern to fix a reference call stack entry
- LAST\_OCCURRENCE\_OF: Uses the last occurrence of a given call stack line pattern to fix a reference call stack entry

Each extraction is applied in either a `TO` or `FROM` manner:

- TO: Extracts that part of the call stack that starts from the top of the call stack to the determined reference call stack entry
- FROM: Extracts that part of the call stack that starts from the determined reference call stack entry to the bottom of the call stack

To illustrate these concepts, consider the following extractions where `LO_CALL_STACK` represents the call stack from the above table:

## ↔ Sample Code

```

DATA(lo_line_pattern) = xco_cp_call_stack->line_pattern->method(
 )->where_class_name_starts_with( 'CL_XCO_ ' ).

DATA(lo_extracted_call_stack) = lo_call_stack->from->position( 2
 )->to->last_occurrence_of( lo_line_pattern
 )->to->position( -2 ).
```

In total three extractions are applied. The first extraction, `->FROM->POSITION( 2 )`, will produce the following call stack:

Line	Positive position	Negative position
CL_XCO_ADT_CR_SIMPLE_ACTION execute [method]	1	-16
CL_XCO_DP_ACTION_ABSTRACT if_xco_dp_action~execute [method]	2	-15
CL_XCO_ADT_CALL_CLASSRUN if_xco_rt_adt_call_classrun~dispatch [method]	3	-14
CL_XCO_RT_ADT_CLASSRUN_ABSTR if_oo_adt_classrun~main [method]	4	-13
CL_OO_ADT_RES_CLASSRUN exe- cute_clas [method]	5	-12
CL_OO_ADT_RES_CLASSRUN post [method]	6	-11
CL_ADT_REST_RESOURCE if_rest_han- dler~handle [method]	7	-10
CL_REST_ROUTER if_rest_handler~han- dle [method]	8	-9
CL_REST_ROUTER if_rest_handler~han- dle [method]	9	-8
CL_REST_HTTP_HANDLER if_http_ex- tension~handle_request [method]	10	-7
CL_ADT_WB_RES_APP lif_request_han- dler~handle_request [method]	11	-6
CL_ADT_WB_RES_APP handle_request [method]	12	-5
CL_ADT_WB_RES_APP if_http_exten- sion~handle_request	13	-4
CL_HTTP_SERVER [system] exe- cute_request [method]	14	-3
SAPLHTTP_RUNTIME [system] http_dispatch_request [function]	15	-2
SAPMHTTP [system] %_http_start [module (pbo)]	16	-1

The second extraction, `->to->last_occurrence_of( lo_line_pattern )`, applied to the result of the first extraction, will then produce the following call:

Line	Positive position	Negative position
CL_XCO_ADT_CR_SIMPLE_ACTION execute [method]	1	-4
CL_XCO_DP_ACTION_ABSTRACT if_xco_dp_action~execute [method]	2	-3
CL_XCO_ADT_CALL_CLASSRUN if_xco_rt_adt_call_classrun~dispatch [method]	3	-2
CL_XCO_RT_ADT_CLASSRUN_ABSTR if_oo_adt_classrun~main [method]	4	-1

Finally, the application of the third extraction, `->to->position( -2 )`, to the result of the second extraction yields the following call stack:

Line	Positive position	Negative position
CL_XCO_ADT_CR_SIMPLE_ACTION execute [method]	1	-3
CL_XCO_DP_ACTION_ABSTRACT if_xco_dp_action~execute [method]	2	-2
CL_XCO_ADT_CALL_CLASSRUN if_xco_rt_adt_call_classrun~dispatch [method]	3	-1

Note that the reference call stack entries are always treated inclusively, i.e. they will be included in the extracted call stack.

## Formatting a call stack

To be written to the console or into an application log (using `IF_XCO_CP_BAL_LOG`), a given call stack can be formatted which will produce an `IF_XCO_TEXT` for the call stack. As of now, the following formats are supported:

- ADT: The ADT format will render a call stack in the same style as the ADT Debugger. Optionally, the captured line numbers (if present) can be added to each resulting call stack entry line whereby source or include based line number flavors are available for selection

Using the call stack from the beginning, formatting it with source-based line numbers and writing the resulting `IF_XCO_TEXT` to the console of an ADT classrun (obtained by inheriting from `CL_XCO_CP_ADT_SIMPLE_CLASSRUN`) can be accomplished as follows:

### ↔ Sample Code

```
DATA(lo_format) = xco_cp_call_stack=>format->adt(
    )->with_line_number_flavor( xco_cp_call_stack=>line_number_flavor->source ).
```

```
DATA(lo_call_stack_text) = lo_call_stack->as_text( lo_format ).  
out->write( lo_call_stack_text ).
```

This will produce the following output on the console (line numbers may vary on local execution):

```
ZCL_XCO_CALL_STACK main [method] Source line: 10  
CL_XCO_ADT_CR_SIMPLE_ACTION execute [method] Source line: 39  
CL_XCO_DP_ACTION_ABSTRACT if_xco_dp_action~execute [method] Source line: 59  
CL_XCO_ADT_CALL_CLASSRUN if_xco_rt_adt_call_classrun~dispatch [method] Source line:  
30  
CL_XCO_RT_ADT_CLASSRUN_ABSTR if_oo_adt_classrun~main [method] Source line: 36  
CL_OO_ADT_RES_CLASSRUN execute_clas [method] Source line: 206  
CL_OO_ADT_RES_CLASSRUN post [method] Source line: 395  
CL_ADT_REST_RESOURCE if_rest_handler~handle [method] Source line: 150  
CL_REST_ROUTER if_rest_handler~handle [method] Source line: 325  
CL_REST_HTTP_HANDLER if_http_extension~handle_request [method] Source line: 190  
CL_ADT_WB_RES_APP lif_request_handler~handle_request [method] Source line: 19  
CL_ADT_WB_RES_APP handle_request [method] Source line: 317  
CL_ADT_WB_RES_APP if_http_extension~handle_request [method] Source line: 435  
CL_HTTP_SERVER [system] execute_request [method] Source line: 2840  
SAPLHTTP_RUNTIME [system] http_dispatch_request [function] Source line: 1626  
SAPMHTTP [system] %_http_start [module (pbo)] Source line: 12
```

## 4.2.11.16.4.4 JSON

The goal of the XCO JSON module is to make working with JSON data as simple as possible. This includes the creation of JSON strings (both from ABAP data structures and from scratch) as well as the translation of JSON strings to ABAP.

### Reading JSON strings

The following code shows an example of how a JSON string (e.g. the response of an outbound service call) can be translated into a corresponding ABAP structure:

#### Sample Code

```
DATA(lv_json_string) = '{'  
  && | "SessionId":"7cd44fff-036a-4155-b0d2-f5a4dfbcee92", |  
  && | "UserName":"John Doe", |  
  && | "IsPremiumUser":true |  
  && '}'.  
TYPES:  
BEGIN OF ts_input,
```

```

        session_id      TYPE sysuuid_c36,
        user_name       TYPE string,
        is_premium_user TYPE abap_bool,
    END OF ts_input.
DATA ls_input TYPE ts_input.
" After execution, the structure components will have the following values
"   session_id = 7cd44fff-036a-4155-b0d2-f5a4dfbcee92
"   user_name = John Doe
"   is_premium_user = X
xco_cp_json=>data->from_string( lv_json_string )->apply( VALUE #(
    ( xco_cp_json=>transformation->pascal_case_to_underscore )
    ( xco_cp_json=>transformation->boolean_to_abap_bool )
) )->write_to( REF #( ls_input ) ).
```

To make it easy to adjust JSON data to ABAP requirements (e.g. to switch between camel case and underscore notation) the XCO library provides built-in transformations which are accessible via the XCO\_C\_P\_JSON API. The current version of the XCO library comes with the following three built-in transformations:

- Camel case/Pascal case to underscore: Transforms the name of each object member in the JSON data from Camel case/Pascal case to underscore notation
- Underscore to Camel case/Pascal case: Transforms the name of each object member in the JSON data from underscore to Camel case/Pascal case notation
- Boolean to abap\_bool: Transforms each JSON boolean value to its abap\_bool equivalent, i.e. true becomes ‘X’ and false becomes ‘

Note that the XCO library uses the following terminology:

- Camel case: The first character of each word is capitalized except for the first word.
- Pascal case: The first character of each word is capitalized including the first word.

## Creating JSON strings

JSON strings can be created in two ways, either from scratch or from a corresponding ABAP data structure. Using the XCO JSON data builder, JSON data can be built like:

### ↳ Sample Code

```

DATA(lo_json_builder) = xco_cp_json=>data->builder( ).
lo_json_builder->begin_object(
    )->add_member( 'SessionId' )->add_string( '7cd44fff-036a-4155-b0d2-
f5a4dfbcee92' )
    )->add_member( 'UserName' )->add_string( 'John Doe'
    )->add_member( 'IsPremiumUser' )->add_boolean( abap_true
    )->end_object( ).
" After execution LV_JSON_STRING will have the value
" { "SessionId": "7cd44fff-036a-4155-b0d2-f5a4dfbcee92", "UserName": "John
Doe", "IsPremiumUser": true }
DATA(lv_json_string) = lo_json_builder->get_data( )->to_string( ).
```

In the same way that a JSON string can be translated to an appropriate ABAP data structure an ABAP data structure can be translated to a JSON string. It is again possible to apply transformations to e.g. transform underscore to camel case notation.

### ↳ Sample Code

```

TYPES:
BEGIN OF ts_output,
    session_id      TYPE sysuuid_c36,
```

```

        user_name      TYPE string,
        is_premium_user TYPE xsdboolean,
    END OF ts_output.
DATA(ls_output) = VALUE ts_output(
    session_id      = '7cd44fff-036a-4155-b0d2-f5a4dfbcee92'
    user_name       = |John Doe|
    is_premium_user = abap_true
).
" After execution LV_JSON_STRING will have the value
" { "SessionId": "7cd44fff-036a-4155-b0d2-f5a4dfbcee92", "UserName": "John
Doe", "IsPremiumUser": true}
DATA(lv_json_string) = xco_cp_json=>data->from_abap( ls_output )-
>apply( VALUE #(
    ( xco_cp_json=>transformation->underscore_to_pascal_case )
) )->to_string( ).
```

## 4.2.11.16.4.5 Regular Expression

Regular expressions can be used to match strings against patterns or extract substrings that match certain criteria.

The regular expression engines offered by XCO are based on the standards offered by CL\_ABAP\_REGEX:

- POSIX
- PCRE
- XPATH2
- XSD

In XCO, these regular expression standards are encapsulated in the IF\_XCO\_REGEX\_ENGINE abstraction which is accessible via the XCO\_CP\_REGULAR\_EXPRESSION API:

### ↳ Sample Code

```

" Example for a POSIX regular expression engine (More configuration options
are available
" as optional parameters of the method POSIX).
DATA(lo_posix_engine) = xco_cp_regular_expression=>engine->posix(
    iv_ignore_case = abap_true
).
" Example for a PCRE regular expression engine (More configuration options
are available
" as optional parameters of the method PCRE).
DATA(lo_pcre_engine) = xco_cp_regular_expression=>engine->pcre(
    iv_ignore_case      = abap_false
    iv_enable_multiline = abap_false
).
" Example for an XPATH 2.0 regular expression engine (More configuration
options are available
" as optional parameters of the method XPATH2).
DATA(lo_xpath2_engine) = xco_cp_regular_expression=>engine->xpath2(
    iv_ignore_case      = abap_true
    iv_enable_multiline = abap_false
).
" Example for an XSD regular expression engine (More configuration options
are available
" as optional parameters of the method XSD).
DATA(lo_xsd_engine) = xco_cp_regular_expression=>engine->xsd(
    iv_ignore_case      = abap_true
    iv_enable_multiline = abap_false
```

). .

The parameters of the regular expression engine factory methods can be used to configure the behavior of the engine when it is used against a string. The default engine (which is used when no explicit engine is supplied) is POSIX in the default configuration as provided by CL\_ABAP\_REGEX.

A regular expression can be used to easily check if a given string matches a pattern

#### ↳ Sample Code

```
" LV_CASE_INSENSITIVE_MATCHES will have the value abap_true as
" LO_POSIX_ENGINE is case insensitive (see above).
DATA(lv_case_insensitive_matches) = xco_cp=>string( |abc| )->matches(
    iv_regular_expression = '^a.*C$'
    io_engine             = lo_posix_engine
).
" LV_CASE_SENSITIVE_MATCHES will have the value abap_false as the
" default POSIX engine is case sensitive.
DATA(lv_case_sensitive_matches) = xco_cp=>string( |abc| )-
>matches( '^a.*C$' ).
```

or to retrieve certain parts from it

#### ↳ Sample Code

```
DATA(lo_name_parts) = xco_cp=>string( |John Doe| )->grep( '^(\S+)\s+(\S+)$' ).
" LV_FIRST_NAME = John
DATA(lv_first_name) = lo_name_parts->value[ 1 ].
" LV_LAST_NAME = Doe
DATA(lv_last_name) = lo_name_parts->value[ 2 ].
```

## 4.2.11.16.4.6 String

Besides providing a straightforward integration with regular expressions (see [Regular Expression \[page 1802\]](#)), the XCO string abstraction offers further simplifications when working with strings.

### The methods TO and FROM

The methods TO and FROM provide flexible ways to extract a substring from a string. Their functionality is based on the following character indexing scheme (The table is based on the example string ABCDEF):

A	B	C	D	E	F
1	2	3	4	5	6
-6	-5	-4	-3	-2	-1

Both TO and FROM have an inclusive behavior, i.e. the provided index is always included in the retrieved substring. Furthermore, if the provided index is out of bounds the behavior is as if the index had pointed to the first (resp. last) character of the string.

The following sample illustrates several usages:

#### ↳ Sample Code

```
DATA(lo_string) = xco_cp=>string( |ABCDEF| ).  
" LV_SUBSTRING_1 = DEF  
DATA(lv_substring_1) = lo_string->from( 4 )->value.  
" LV_SUBSTRING_2 = DEF  
DATA(lv_substring_2) = lo_string->from( -3 )->value.  
" LV_SUBSTRING_3 = ABC  
DATA(lv_substring_3) = lo_string->to( 3 )->value.  
" LV_SUBSTRING_4 = ABC  
DATA(lv_substring_4) = lo_string->to( -4 )->value.  
" LV_SUBSTRING_5 = CD  
DATA(lv_substring_5) = lo_string->from( 3 )->to( 2 )->value.  
" LV_SUBSTRING_6 = BC  
DATA(lv_substring_6) = lo_string->to( -4 )->from( 2 )->value.
```

## Splitting and joining

The XCO string library also provides a way to split strings and join a list of strings:

#### ↳ Sample Code

```
DATA(lo_name_parts) = xco_cp=>string( |John Doe| )->split( | | ).  
" LV_REVERSED_NAME = Doe, John  
DATA(lv_reversed_name) = lo_name_parts->reverse( )->join( |, | )->value.
```

## Composing and decomposing

A generalization of split and join operations on strings are string compositions and decompositions.

String compositions and decompositions encapsulate algorithms which can transform a string to a list of strings (decomposition) and vice versa (composition) according to a certain formula. Currently, both Camel Case and Pascal Case are supported in both directions.

The following example illustrates how the camel case composition and decomposition can be used to easily translate between underscore and camel case notation.

#### ↳ Sample Code

```
DATA(lv_string_with_underscores) = |FIRST_NAME| .  
" LV_STRING_IN_CAMEL_CASE = firstName  
DATA(lv_string_in_camel_case) = xco_cp=>string( lv_string_with_underscores  
)->split( |_|  
)->compose( xco_cp_string=>composition->camel_case  
)->value.  
" LV_RESTORED_STRING = FIRST_NAME  
DATA(lv_restored_string) = xco_cp=>string( lv_string_in_camel_case  
)->decompose( xco_cp_string=>decomposition->camel_case  
)->join( |_|  
)->to_upper_case(  
)->value.
```

## Converting strings to xstrings

Converting a given string to its xstring representation for a given code page can be easily accomplished by making use of `IF_XCO_CHAR_CODE_PAGES` which are available via `XCO_CP_CHARACTER`:

#### ↔ Sample Code

```
DATA(lv_string) = |ABC|.  
" The representation of LV_STRING as an XSTRING where the given character  
" code page is used for the conversion.  
DATA(lv_xstring) = xco_cp=>string( lv_string  
)->as_xstring( xco_cp_character=>code_page->utf_8 ).
```

## Calculating Hash Codes

Calculating the hash value of a given string for a given hash algorithm can be accomplished as follows:

#### ↔ Sample Code

```
" Upon execution, LV_HASH_VALUE will have the value  
C79C561BB2CC3D0430A54B3D014C89C3089FE089.  
DATA(lv_hash_value) = xco_cp=>string( 'MY_STRING'  
)->as_xstring( xco_cp_hash=>algorithm->sha_1  
)->value.
```

Note the following details:

- Supported hash algorithms (represented as objects of type IF\_XCO\_HASH\_ALGORITHM) can be obtained via XCO\_CP\_HASH=>ALGORITHM. The SHA-1 hash algorithm is statically available via the SHA\_1 property while the FOR method can be used to obtain hash algorithms based on their string identifiers, e.g. XCO\_CP\_HASH=>ALGORITHM→FOR( 'MD5' ) for the MD5 hash algorithm
- When calculating the hash value of a string, an IF\_XCO\_HASH\_ALGORITHM object acts as an IF\_XCO\_STRING\_XSTRING\_CNVRSN, where the XSTRING that is derived from the string corresponds to the hash value

## 4.2.11.16.4.7 System

With the XCO system module, you can easily request information about the current system.

## Namespaces

You can get a handle for a given namespace via the API xco\_cp\_system=>NAMESPACE. Once obtained, you can use a namespace handle to find out whether the given namespace exists in the system and, if so, whether it's changeable:

#### ↔ Sample Code

```
" LO_NAMESPACE is a handle that can be used to request information  
" about namespace /ABC/.  
DATA(lo_namespace) = xco_cp_system=>namespace->for( '/ABC/' ).
```

```

" LV_NAMESPACE is an ABAP_BOOL value indicating whether the namespace
" represented by LO_NAMESPACE exists or not.
DATA(lv_namespace_exists) = lo_namespace->exists( ).

" LV_NAMESPACE_IS_CHANGEABLE is an ABAP_BOOL value indicating whether the
" namespace represented by LO_NAMESPACE is changeable or not. Note that,
" before invoking this method, it should be ensured that the namespace exists
" as otherwise a runtime exception will be raised.
DATA(lv_namespace_is_changeable) = lo_namespace->is_changeable( ).
```

## 4.2.11.16.4.8 Tenant

The XCO tenant module provides standard abstractions for querying information about tenants, in particular the tenant associated with the current ABAP session.

The central abstraction `IF_XCO_CP_TENANT` represents a tenant and is used to retrieve its public information. As of now, the following information can be retrieved:

- ID: The ID for the tenant
- GUID: The GUID for the tenant (represented as an `IF_XCO_UUID`)
- URL: The URL of the tenant for the given URL type (URL types are available via `XCO_CP_TENANT=>URL_TYPE`)
- Global Account ID: The ID of the global account associated with the tenant
- Subaccount ID: The ID of the subaccount associated with the tenant

The currently active tenant can be obtained via:

```
DATA(lo_current_tenant) = xco_cp->current->tenant( ).
```

Note that depending on the overall system infrastructure and setup a current tenant doesn't necessarily exist, which would be indicated by an initial reference returned by the statement above. Once the current tenant has been obtained by the URL (in the example below for the type UI) along with its different components, the currently active tenant can be obtained as follows:

### ↔ Sample Code

```

" UI URL of the current tenant.
DATA(lo_ui_url) = lo_current_tenant->get_url( xco_cp_tenant=>url_type->ui ).

" The protocol of the UI URL (Type string).
DATA(lv_ui_url_protocol) = lo_ui_url->get_protocol( ).

" The host (including the domain) of the UI URL (Type string).
DATA(lv_ui_url_host) = lo_ui_url->get_host( ).

" The port of the UI URL (Type i).
DATA(lv_ui_url_port) = lo_ui_url->get_port( ).
```

In a similar way it's also possible to access the information about the associated global account and subaccount:

#### ↳ Sample Code

```
" The string value of the global account ID of the current tenant.  
DATA(lv_global_account_id) = lo_current_tenant->get_global_account_id(  
    )->as_string( ).  
" The string value of the subaccount ID of the current tenant.  
DATA(lv_subaccount_id) = lo_current_tenant->get_subaccount_id(  
    )->as_string( ).
```

### 4.2.11.16.4.9 Time Library

The XCO time library provides standard abstractions for working with temporal values.

- A date (IF\_XCO\_CP\_TM\_DATE) consists of values for year, month and day
- A time (IF\_XCO\_CP\_TM\_TIME) consists of values for hour, minute and second
- A moment (IF\_XCO\_CP\_TM\_MOMENT) consists of a fully specified date and time
- A UNIX timestamp (IF\_XCO\_CP\_TM\_UNIX\_TIMESTAMP) consists of an INT8 value specifying the number of seconds that have elapsed since 00:00:00 UTC on Jan 1st, 1970.

#### Dates, times and moments

The abstractions date, time and moment correspond to the ABAP temporal types DATS, TIMS and TZNTSTMPs. Given a date, time or moment object, a new object can be derived from each of them by adding or subtracting values from their components or directly overriding specific values. Note that date, time and moment objects are always immutable, i.e. once constructed their values will never change.

Time zone information is explicitly not part of the state of a date, time or moment object.

#### Time formats

A time format is a unique representation of a date, time or moment. Currently, the following time formats are supported:

- ABAP: Will provide the DATS, TIMS and TZNTSTMPs value, respectively
- ISO 8601 Basic
- ISO 8601 Extended

Time formats are accessible via XCO\_CP\_TIME=>FORMAT and can be used either directly or passed to the AS method of a date, time or moment object to obtain the string representation of the object according to the provided time format.

The following code sample illustrates the usage of the various time formats for a moment object:

#### ↳ Sample Code

```
DATA(lo_moment) = xco_cp_time=>moment(  
    iv_year    = '2020'  
    iv_month   = '09'  
    iv_day     = '16'  
    iv_hour    = '22'  
    iv_minute  = '33'  
    iv_second  = '49'  
).  
" LV_ABAP_FORMAT will be of type STRING and have the value 20200916223349.
```

```

DATA(lv_abap_format) = lo_moment->as( xco_cp_time=>format->abap )->value.
" LV_ISO_8601_BASIC will be of type STRING and have the value 20200916T223349.
DATA(lv_iso_6801_basic) = lo_moment->as( xco_cp_time=>format-
>iso_8601_basic )->value.
" LV_ISO_8601_EXTENDED will be of type STRING and have the value
2020-09-16T22:33:49.
DATA(lv_iso_6801_extended) = lo_moment->as( xco_cp_time=>format-
>iso_8601_extended )->value.
" Restore the moment from LV_ISO_8601_EXTENDED. LO_RESTORED_MOMENT will be an
object of type
" IF_XCO_CP_TM_MOMENT with values equal to LO_MOMENT.
DATA(lo_restored_moment) = xco_cp_time=>format->iso_8601_extended-
>to_moment( lv_iso_6801_extended ).
```

## Integration with XCO\_CP=>SY

The XCO time library is tightly integrated with the XCO abstraction for the ABAP SY structure to provide an easy way to access the current time, either as a date, time or moment object or as a UNIX timestamp.

When a date, time or moment object is obtained, the time zone according to which the values should be calculated can be specified in addition. Note that a UNIX timestamp will always be in UTC.

### ↔ Sample Code

```

" The current moment in the time zone of the active user.
DATA(lo_moment_user) = xco_cp=>sy->moment( xco_cp_time=>time_zone->user ).
" The current moment in UTC.
DATA(lo_moment_utc) = xco_cp=>sy->moment( xco_cp_time=>time_zone->utc ).
" The current UNIX timestamp.
DATA(lo_unix_timestamp) = xco_cp=>sy->unix_timestamp( ).
```

Note that if a time zone is not explicitly specified when a date, time or moment object is retrieved from XCO\_CP=>SY, the time zone of the active user will be taken.

## UNIX timestamps

UNIX timestamps are a common way for describing a given point in time by denoting the number of elapsed seconds since 00:00:00 UTC on Jan 1st, 1970. The XCO time library makes it easy to convert from a given UNIX timestamp to a moment and vice versa:

### ↔ Sample Code

```

" The current UNIX timestamp.
DATA(lo_unix_timestamp) = xco_cp=>sy->unix_timestamp( ).
" Get the moment for the UNIX timestamp (in the time zone of the active
" user).
DATA(lo_moment) = lo_unix_timestamp->get_moment( xco_cp_time=>time_zone-
>user ).
" Restore the original UNIX timestamp. The value of
LO_RESTORED_UNIX_TIMESTAMP is
" equal to that of LO_UNIX_TIMESTAMP.
DATA(lo_restored_unix_timestamp) = lo_moment-
>get_unix_timestamp( xco_cp_time=>time_zone->user ).
```

## 4.2.11.16.4.10 UUID

To effectively work with UUIDs the XCO standard library provides a simple way to translate between different UUID formats:

### ↔ Sample Code

```
DATA(lo_uuid) = xco_cp_uuid=>format->c36->to_uuid( '7cd44fff-036a-4155-b0d2-f5a4dfbcee92' ).  
" LV_UUID_C32 will hold the value 7CD44FFF036A4155B0D2F5A4DFBCEE92  
DATA(lv_uuid_c32) = CONV sysuuid_c32( xco_cp_uuid=>format->c32->from_uuid( lo_uuid ) ).
```

## 4.2.11.16.4.11 XLSX

The XCO XLSX module offers a set of standard abstractions and APIs to programmatically work with XLSX workbooks and their worksheets (e.g. coming from an uploaded Microsoft Excel .XLSX file).

The XCO XLSX module exposes three external APIs:

- XCO\_CP\_XLSX: Access to general abstractions used throughout the XCO XLSX module
- XCO\_CP\_XLSX\_SELECTION: Access to selection pattern builders used to build selection patterns
- XCO\_CP\_XLSX\_READ\_ACCESS: Access to abstractions used when reading the content of an XLSX worksheet

## Document handles

The starting point for any programmatic interaction with an XLSX workbook and its worksheets is obtaining a handle for the containing document via XCO\_CP\_XLSX=>DOCUMENT. As of now, document handles can be obtained for the complete file content of an .XLSX file, which must be provided as an XSTRING:

### ↔ Sample Code

```
DATA lv_file_content TYPE xstring.  
  
" LV_FILE_CONTENT must be populated with the complete file content of  
the .XLSX file  
" whose content shall be processed programmatically.  
  
DATA(lo_document) = xco_cp_xlsx=>document->for_file_content( lv_file_content ).
```

The obtained document handle then allows to obtain a read access, which is the starting point for accessing the cell contents of the worksheets contained in the document [XLSX Read Access \[page 1812\]](#).

## Coordinates

Within the XCO XLSX module, a worksheet is regarded as a space with two dimensions, with the first dimension being the column and the second dimension being the row. Consequently, a cell in a worksheet is uniquely determined by providing values for its column and row. A coordinate (an object of type CL\_XCO\_XLSX\_COORDINATE) uniquely fixes a value for a given dimension, for example, for a value for a column or row. In office suites, e.g. Microsoft Excel, alphabetic values are used to identify columns and numeric values are used to identify rows. The XCO XLSX module provides support for both variants so that alphabetic as well as numeric values can be used freely when values are provided for columns and rows.

- Alphabetic coordinates: A, B, C, ... Z, AA, AB, ...ZZ, AAA, AAB, ...
- Numeric coordinates: 1, 2, 3, ...

Coordinates can be obtained via XCO\_CP\_XLSX=>COORDINATE for both alphabetic and numeric values and once obtained, both the alphabetic as well as the numeric value can be retrieved for a coordinate. This also provides an easy way to translate between alphabetic and numeric coordinate values, for example:

### ↔ Sample Code

```
DATA(lo_coordinate_1) = xco_cp_xlsx=>coordinate->for_alphabetic_value( 'AA' ).  
" LV_COORD_1_NUMERIC_VALUE will be an integer with value 27.  
DATA(lv_coord_1_numeric_value) = lo_coordinate_1->get_numeric_value( ).  
" LV_COORD_1_ALPHABETIC_VALUE will be a string with value AA.  
DATA(lv_coord_1_alphabetic_value) = lo_coordinate_1->get_alphabetic_value( ).  
DATA(lo_coordinate_2) = xco_cp_xlsx=>coordinate->for_numeric_value( 27 ).  
" LV_COORD_2_NUMERIC_VALUE will be an integer with value 27.  
DATA(lv_coord_2_numeric_value) = lo_coordinate_2->get_numeric_value( ).  
" LV_COORD_2_ALPHABETIC_VALUE will be a string with value AA.  
DATA(lv_coord_2_alphabetic_value) = lo_coordinate_2->get_alphabetic_value( ).
```

It's possible to derive new coordinates from existing coordinates such as objects of type CL\_XCO\_XLSX\_COORDINATE, by using the SHIFT method to either increment or decrement the value of the coordinate (depending on the sign of the provided offset).

## Selection patterns

Selections are a central concept within the XCO XLSX module. A selection is based on a selection pattern which encapsulates a prescription according to which individual cells are identified in a given worksheet which then form the selection. To illustrate the concepts and offered selection patterns consider the following exemplary portion of an XLSX worksheet:

A	B	C	D	E	F	...
1						
2		1	2	3	4	

A	B	C	D	E	F	...
3	5	6	7	8		
4	9	10	11	12		
5		13	14	15	16	
6						
...						

For the purposes of the following code samples, it's assumed that except for the cell values shown above, all other cells of the worksheet are initial. Supported selection patterns can be built via selection pattern builders which are obtainable via XCO\_CP\_XLSX\_SELECTION=>PATTERN\_BUILDER.

## Simple from-to selection patterns

A simple from-to selection pattern allows to define static boundary values for columns and rows. It's not required to specify a boundary value for each dimension, hence both bounded and unbounded selections are possible. Consider the following characteristic examples:

### Sample Code

```
" The selection pattern LO_PATTERN_1 will be a bounded rectangle, matching
values
" 6, 7, 10 and 11 of the above worksheet.
DATA(lo_pattern_1) = xco_cp_xlsx_selection->pattern_builder->simple_from_to(
    )->from_column( xco_cp_xlsx=>coordinate->for_alphabetic_value( 'C' ) )
    )->to_column( xco_cp_xlsx=>coordinate->for_alphabetic_value( 'D' ) )
    )->from_row( xco_cp_xlsx=>coordinate->for_numeric_value( 3 ) )
    )->to_row( xco_cp_xlsx=>coordinate->for_numeric_value( 4 ) )
    )->get_pattern( ).

" The selection pattern LO_PATTERN_2 will be unbounded at the bottom, matching
" values 6, 7, 10, 11, 14 and 15 of the above worksheet.
DATA(lo_pattern_2) = xco_cp_xlsx_selection->pattern_builder->simple_from_to(
    )->from_column( xco_cp_xlsx=>coordinate->for_alphabetic_value( 'C' ) )
    )->to_column( xco_cp_xlsx=>coordinate->for_alphabetic_value( 'D' ) )
    )->from_row( xco_cp_xlsx=>coordinate->for_numeric_value( 3 ) )
    )->get_pattern( ).

" The selection pattern LO_PATTERN_3 will be unbounded at the bottom and at
the
" right, matching values 6, 7, 8, 10, 11, 12, 14, 15 and 16 of the above
worksheet.
DATA(lo_pattern_3) = xco_cp_xlsx_selection->pattern_builder->simple_from_to(
    )->from_column( xco_cp_xlsx=>coordinate->for_alphabetic_value( 'C' ) )
    )->from_row( xco_cp_xlsx=>coordinate->for_numeric_value( 3 ) )
    )->get_pattern( ).

" The selection pattern LO_PATTERN_4 is completely unbounded, matching all
cells of the
" worksheet, hence also all values of the above worksheet.
DATA(lo_pattern_4) = xco_cp_xlsx_selection->pattern_builder->simple_from_to(
    )->get_pattern( ).
```

## 4.2.11.16.4.11.1 XLSX Read Access

The starting point for programmatically reading the content of an XLSX document is to get a read access for the document. Find out how this is done.

### ⚠ Caution

Note that when data is read from a given XLSX document via the XCO XLSX module, the data is provided in exactly the way it's stored within the XLSX document. This implies that no security or any other kind of additional validations are performed against the data contained in the worksheets of the XLSX workbook when it's accessed by the means described below. Please ensure that when the read data is further processed by application logic (for example, stored in a database table or shown in a Fiori app), dedicated checks are in place to guard your application against potentially malicious content (such as string values in an XLSX worksheet containing JavaScript code).

Consider the following:

### ↔ Sample Code

```
DATA lv_file_content TYPE xstring.  
  
" LV_FILE_CONTENT must be populated with the complete file content of  
the .XLSX file  
" whose content will be processed programmatically.  
  
DATA(lo_read_access) = xco_cp_xlsx=>document-  
>for_file_content( lv_file_content  
 )->read_access( ).
```

Once obtained, the next step is to get the read access for the worksheet which contains the data that will be read. It's possible to identify a worksheet based on its name or position:

### ↔ Sample Code

```
" Read access for the worksheet at position 1, such as the first worksheet in  
the workbook.  
DATA(lo_first_worksheet) = lo_read_access->get_workbook(  
 )->worksheet->at_position( 1 ).  
  
" Read access for the worksheet with name INVOICES.  
DATA(lo_invoices_worksheet) = lo_read_access->get_workbook(  
 )->worksheet->for_name( 'INVOICES' ).
```

## Accessing Data via a Stream

The first way to gain access to the data stored in a worksheet is by selecting a collection of cells based on a selection pattern (see [XLSX \[page 1809\]](#)=>*Selection patterns*). You can do this using the method `SELECT` on `IF_XCO_XLSX_RA_WORKSHEET`. You can access the cells contained in the selection following a stream-based approach, meaning that you'll get a dedicated stream for the selection which will provide sequential access to the individual blocks of the selection. Two kinds of streams are offered:

- Cell stream: A cell stream provides access to the selection one cell at a time, traversing the selection from left to right and top to bottom
- Row stream: A row stream provides access to the selection one row at a time, traversing the selection from top to bottom

## Cell streams

Cell streams are intended for dynamic reading scenarios where it's required to process each cell individually. The following operations are offered for cell streams:

- Visit: The visit operation takes a visitor implementation as the input (an object of type `IF_XCO_XLSX_RA_CS_VISITOR`) which defines the logic that should be performed for each cell. This construction follows the `Visitor` design pattern.

Consider the following example of how a visit operation can be obtained and run:

### → Sample Code

```
" A selection pattern that was obtained via
XCO_CP_XLSX_SELECTION=>PATTERN_BUILDER.
DATA lo_selection_pattern TYPE REF TO if_xco_xlsx_slc_pattern.

" The read access to the worksheet.
DATA lo_worksheet TYPE REF TO if_xco_xlsx_ra_worksheet.

" The implementation of the visitor, such as a dedicated local class
containing the
" logic that will be performed for each visited cell.
DATA lo_visitor TYPE REF TO if_xco_xlsx_ra_cs_visitor.

lo_worksheet->select( lo_selection_pattern
    )->cell_stream(
        )->operation->visit( lo_visitor
        )->if_xco_xlsx_ra_operation->execute( ).
```

## Row streams

Use row streams when the structure of the data that should be read is statically known. The primary use case is to read a portion of a worksheet (as identified by a selection) into an internal table. The following operations are offered for row streams:

- Write To: The write to operation takes a reference to an internal table as the input to which the selected data will be written (when running the operation)

Consider the following example of how a write to operation can be obtained and run:

### → Sample Code

```
" A selection pattern that was obtained via
XCO_CP_XLSX_SELECTION=>PATTERN_BUILDER.
DATA lo_selection_pattern TYPE REF TO if_xco_xlsx_slc_pattern.

" The read access to the worksheet.
DATA lo_worksheet TYPE REF TO if_xco_xlsx_ra_worksheet.

" The type definition resembling the structure of the rows in the worksheet
selection.
TYPES:
  BEGIN OF ts_row,
    first_name      TYPE string,
    last_name       TYPE string,
```

```

    day_of_birth TYPE d,
END OF ts_row,

tt_row TYPE STANDARD TABLE OF ts_row WITH DEFAULT KEY.

DATA lt_rows TYPE tt_row.

lo_worksheet->select( lo_selection_pattern
    )->row_stream(
    )->operation->write_to( REF #( lt_rows )
    )->if_xco_xlsx_ra_operation~execute( ).

" At this point, the internal table LT_ROWS will contain the rows from the
worksheet
" selection.

```

## Accessing Data via a Cursor

An alternative to accessing the data in a worksheet via selections and streams is to get a cursor for a worksheet read access using the method CURSOR on IF\_XCO\_XLSX\_RA\_WORKSHEET. Just as with desktop office suites, you can first position a cursor on any given cell (identified by coordinate values for both the column and row of the cell). Afterwards, you can move it around the worksheet freely via the methods on IF\_XCO\_XLSX\_RA\_CURSOR:

- You can use the methods MOVE\_UP, MOVE\_RIGHT, MOVE\_DOWN and MOVE\_LEFT to move the cursor relative to its current position by the given number of steps
- You can use the methods SET\_COLUMN and SET\_ROW to set the new column or row for the cursor
- You can use the method HAS\_CELL to determine if the underlying worksheet contains a cell for the current position of the cursor. If so, you can access it using the method GET\_CELL

The following example illustrates how the string values of the cells in column A starting at row 5 can be read out until the first row is encountered for which the worksheet has no cell or the cell has no value:

### Sample Code

```

" The read access to the worksheet.
DATA lo_worksheet TYPE REF TO if_xco_xlsx_ra_worksheet.

DATA(lo_cursor) = lo_worksheet->cursor(
    io_column = xco_cp_xlsx=>coordinate->for_alphabetic_value( 'A' )
    io_row     = xco_cp_xlsx=>coordinate->for_numeric_value( 5 )
).

WHILE lo_cursor->has_cell( ) EQ abap_true
    AND lo_cursor->get_cell( )->has_value( ) EQ abap_true.
    DATA(lo_cell) = lo_cursor->get_cell( ).

    DATA(lv_string_value) = ``.
    lo_cell->get_value(
        )->set_transformation( xco_cp_xlsx_read_access=>value_transformation-
>string_value
        )->write_to( REF #( lv_string_value ) ).

    " At this point LV_STRING_VALUE contains the string value of the cell
    " at the current position of the cursor.

    " Move the cursor down one row.
    lo_cursor->move_down( ).

```

```
ENDWHILE.
```

## Reading Out Hyperlinks

When you access data using a cursor, it's also possible to read out the hyperlink associated with a given cell. Once you get a handle for a cell in the form of an object of type `IF_XCO_XLSX_RA_CELL`, it's possible to

- Check if a hyperlink is associated to the given cell via the method `HAS_HYPERLINK` on `IF_XCO_XLSX_RA_CELL`
- Get the handle for the hyperlink of the given cell via the method `GET_HYPERLINK` on `IF_XCO_XLSX_RA_CELL`

The handle for a hyperlink, `IF_XCO_XLSX_RA_HYPERLINK`, can then be used to get both the target and the location of the hyperlink via the methods `GET_TARGET` and `GET_LOCATION`. Both target and location are returned exactly as they are stored within the XLSX document.

## Value Transformations

When accessing the value of an individual cell (via `IF_XCO_XLSX_RA_CELL_VALUE`) or of a complete row (as part of a row stream operation), it's possible to apply transformations to the value, which will affect how the cell value is written to an ABAP data field. Technically, a value transformation that you can get using the method `XCO_CP_XLSX_READ_ACCESS=>VALUE_TRANSFORMATION` contains a transformation routine that can be applied to

- Values of individual cells (in case the value transformation implements the interface `IF_XCO_XLSX_RA_VT_CELL_VALUE`)
- Values of rows (in case the value transformation implements the interface `IF_XCO_XLSX_RA_VT_ROW_VALUE`)

The following value transformations are currently offered:

- Identity
- String value
- Best effort

The default value transformation is the best effort value transformation. You can overwrite it using

- The method `SET_VALUE_TRANSFORMATION` of `IF_XCO_XLSX_RA_CELL_VALUE` when the value of an individual cell is read
- The method `SET_VALUE_TRANSFORMATION` of `IF_XCO_XLSX_RA_RS_OP_WRITE_TO` when row values are read and written to an internal table as part of the write to row stream operation

### 'Identity' value transformation

The 'identity' value transformation doesn't apply any modification to the XLSX type or value of a cell. The ABAP field that the cell value will be written to must be fully compliant with the type and value of the cell as it's stored in the XLSX file. The value is first written to a string data object, which is then written to the provided ABAP field without changes. If this write can't be performed successfully, a runtime error appears.

### Note

For floating point numbers, please use one of the `decfloat` built-in types. Don't use packed numbers (type P) since it might lead to wrong conversions.

### 'String value' value transformation

The 'string value' value transformation gets the stringified value of any cell so that it can always be safely written to an ABAP field of type STRING.

### 'Best effort' value transformation

The 'best effort' value transformation is based on an inspection of the ABAP field that a given cell value will be written to. Based on the type determined for the target ABAP field, a transformation is applied to the cell value. The 'best effort' value transformation behaves just as the 'identity' value transformation except for the following types of ABAP fields:

- D: When a cell value is written to an ABAP date field, the value of the cell is interpreted as a date, and it's converted to the ABAP date format
- T: When a cell value is written to an ABAP time field, the value of the cell is interpreted as a time, and it's converted to the ABAP time format
- Data element MSEHI: When a cell value is written to an ABAP field typed against data element MSEHI, the cell value is interpreted as the external value for a unit of measurement, which is converted to the internal ABAP format using conversion routine CUNIT
- Data element SPRAS: When a cell value is written to an ABAP field typed against data element SPRAS, the cell value is interpreted as the external value for a language, which is converted to the internal ABAP format using conversion routine ISOLA

## 4.2.11.16.4.11.2 XLSX Write Access

The starting point for programmatically writing the content of an XLSX document is to get a write access for the document. Find out how this is done.

### Context

You can create a new, empty XLSX document and get write access for it like this:

#### Sample Code

```
DATA(lo_write_access) = xco_cp_xlsx->document->empty( )->write_access( ).
```

An empty XLSX document consists of one worksheet named `sheet1` which you can access via

#### Sample Code

```
DATA(lo_worksheet) = lo_write_access->get_workbook(
```

```
) ->worksheet ->at_position( 1 ).
```

You can populate it with data using the means described below. Once the worksheet has been filled as desired, you can get the corresponding file content of the document as an XSTRING via

#### ↳ Sample Code

```
DATA(lv_file_content) = lo_write_access->get_file_content( ).
```

Adding a new worksheet works like this:

#### ↳ Sample Code

```
DATA(lo_worksheet) = lo_write_access->get_workbook(
    )->add_new_sheet( ).
```

You can provide a name for the sheet by filling the optional parameter `iv_name`.

To modify an existing XSLX document, get a write access like this:

#### ↳ Sample Code

```
DATA(lo_write_access) = xco_cp_xlsx=>document-
    >for_file_content( lv_file_content )->write_access( ).
```

## Writing Data via a Stream

The first way how you can write data into a worksheet is by selecting a collection of cells based on a selection pattern (see also the section [Selection patterns](#) here: [XSLX \[page 1809\]](#)) via the method `SELECT` on `IF_XCO_XLSX_WA_WORKSHEET`. Accessing the cells contained in the selection is done following a stream-based approach, which means that a dedicated stream is obtained for the selection that will provide sequential access to the individual blocks of the selection. One kind of stream is currently offered:

- Row stream: A row stream provides access to the selection one row at a time, traversing the selection from top to bottom

### Row streams

Row streams are best used when the structure of the data that should be written is statically known. The primary use case is to write the rows of an internal table into a corresponding portion of the worksheet (as identified by a selection). The following operations are offered for row streams:

- Write From: The write from operation takes a reference to an internal table as the input whose rows will be written to the selected rows in the worksheet (upon running the operation)

Consider the following example of how a write from operation can be obtained and run:

#### ↳ Sample Code

```
" A selection pattern that was obtained via
XCO_CP_XLSX_SELECTION=>PATTERN_BUILDER.
```

```

DATA lo_selection_pattern TYPE REF TO if_xco_xlsx_slc_pattern.

" The write access to the worksheet.
DATA lo_worksheet TYPE REF TO if_xco_xlsx_wa_worksheet.

" The type definition for the internal table.
TYPES:
BEGIN OF ts_row,
  first_name  TYPE string,
  last_name   TYPE string,
  day_of_birth TYPE d,
END OF ts_row,

tt_row TYPE STANDARD TABLE OF ts_row WITH DEFAULT KEY.

DATA lt_rows TYPE tt_row.

lo_worksheet->select( lo_selection_pattern
  )->row_stream(
  )->operation->write_from( REF #( lt_rows )
  )->execute( ).

" At this point, the rows of internal table LT_ROWS will have been written
into the
" worksheet selection.

```

## Writing Data via a Cursor

An alternative to writing data into a worksheet via selections and streams is to get a cursor for a worksheet write access using the method `CURSOR` on `IF_XCO_XLSX_WA_WORKSHEET`. Just as with desktop office suites, you can position a cursor on any given cell (identified by coordinate values for both the column and row of the cell). Afterwards, you can move it around the worksheet freely via the methods on `IF_XCO_XLSX_WA_CURSOR`:

- You can use the methods `MOVE_UP`, `MOVE_RIGHT`, `MOVE_DOWN` and `MOVE_LEFT` to move the cursor relative to its current position by the given number of steps
- You can use the methods `SET_COLUMN` and `SET_ROW` to set the new column or row for the cursor
- Method `GET_CELL` provides access to the cell at the current position of the cursor

The following example illustrates how the current date and time (stored in ABAP variables `LV_DATE` and `LV_TIME`) can be written into a worksheet:

### ↔ Sample Code

```

" The write access to the worksheet.
DATA lo_worksheet TYPE REF TO if_xco_xlsx_wa_worksheet.

DATA(lo_cursor) = lo_worksheet->cursor(
  io_column = xco_cp_xlsx=>coordinate->for_alphabetic_value( 'B' )
  io_row    = xco_cp_xlsx=>coordinate->for_numeric_value( 2 )
).

" Write the current date.
lo_cursor->get_cell( )->value->write_from( 'Date:' ).

DATA(lv_date) = CONV d( xco_cp=>sy->date( )->as( xco_cp_time=>format->abap )-
>value ).
lo_cursor->move_right( )->get_cell( )->value->write_from( lv_date ).

```

```

" Write the current time.
lo_cursor->move_down( )->move_left( )->get_cell( )->value-
>write_from( 'Time:' ).

DATA(lv_time) = CONV t( xco_cp=>sy->time( )->as( xco_cp_time=>format->abap )-
>value ).
lo_cursor->move_right( )->get_cell( )->value->write_from( lv_time ).
```

For an otherwise empty worksheet, this will produce the following content:

A	B	C
1		
2	Date:	\$LV_DATE\$
3	Time:	\$LV_TIME\$

where \$LV\_DATE\$ is the current date and \$LV\_TIME\$ is the current time.

## Value Transformations

When writing the value of an individual cell (via `IF_XCO_XLSX_WA_CELL_VALUE`) or of a complete row (as part of a row stream operation) it's possible to apply transformations to the value, which will affect what XLSX value is written to the cell in the worksheet. Technically, a value transformation that you can get using the method `xco_cp_xlsx_write_access=>value_transformation` contains a transformation routine that can be applied to

- Values of individual cells (in case the value transformation implements the interface `IF_XCO_XLSX_WA_VT_CELL_VALUE`)
- Values of rows (in case the value transformation implements the interface `IF_XCO_XLSX_WA_VT_ROW_VALUE`)

The following value transformations are currently offered:

- Best effort

The default value transformation is the best effort value transformation. Value transformations can be set explicitly via the following methods:

- Method `SET_VALUE_TRANSFORMATION` of `IF_XCO_XLSX_WA_CELL_VALUE` when the value of an individual cell is written
- Method `SET_VALUE_TRANSFORMATION` of `IF_XCO_XLSX_WA_RS_OP_WRITE_FRM` when row values are written as part of the write from row stream operation

### "Best effort" value transformation

The **best effort** value transformation is based on an inspection (based on ABAP runtime type services) of the ABAP field that should be written to a given cell. Based on the type determined for the ABAP field, a transformation is applied to the value before it's written to the cell of the worksheet. The **best effort** value transformation provides support for the following types of ABAP fields:

- Type `ABAP_BOOL`: When an ABAP field of type `ABAP_BOOL` is written to a cell, a boolean value will be written to the worksheet

- D: When an ABAP field of type D is written to a cell, a date value will be written to the worksheet
- T: When an ABAP field of type T is written to a cell, a time value will be written to the worksheet
- Data element MSEHI: When an ABAP field typed against data element MSEHI is written to a cell, the external value of the corresponding unit of measurement as determined by conversion routine CUNIT will be written to the worksheet
- Data element SPRAS: When an ABAP field typed against data element SPRAS is written to a cell, the external value of the corresponding language as determined by conversion routine ISOLA will be written to the worksheet
- C, N and STRING: When an ABAP field of type C, N or STRING is written to a cell, a string value will be written to the worksheet
- I, INT8 and P: When an ABAP field of type I, INT8 or P is written to a cell, a numeric value will be written to the worksheet

If an attempt is made to write an ABAP field of any other type to a cell of a worksheet using the best effort value transformation, you can expect a runtime error.

## Cell Styling

Cells in an XSLX document can be styled in various ways. To change the background color of a cell, do the following:

### ↔ Sample Code

```
DATA(lo_cursor) = lo_worksheet->cursor(
  io_column = xco_cp_xlsx->coordinate->for_alphabetic_value( 'B' )
  io_row     = xco_cp_xlsx->coordinate->for_numeric_value( 2 )
).
DATA(lo_cell) = lo_cursor->get_cell( ).
DATA(lo_fill_color_yellow) = xco_cp_xlsx->style->fill( )-
>set_background_color( xco_cp_xlsx->color->standard->yellow ).
lo_cell->apply_styles( VALUE #( ( lo_fill_color_yellow ) ) ).
```

You can also protect cells from modifications by locking the entire worksheet. Single cells can then be unlocked to allow selective modifications. See the following example:

### ↔ Sample Code

```
lo_worksheet->protect( ).
DATA(lo_protection_unlock_cell) = xco_cp_xlsx->style->protection( )-
>set_locked( abap_false ).
lo_cell->apply_styles( VALUE #( ( lo_protection_unlock_cell ) ) ).
```

To show a drop-down list of values for a cell, set the data validation. Add the values that should be shown on the list by calling add\_source with a string that contains a single value or a comma-separated list of values. See the following example:

### ↔ Sample Code

```
DATA(lo_data_validation) = xco_cp_xlsx->data_validation_type-
>if_xco_xlsx_dat_val_type_f~list( )->add_source( '1'
 )->add_source( '2' )
```

```
    )->add_source( '3,4,5' ).  
lo_cell->data_validation->set_type( lo_data_validation ).
```

## 4.2.11.16.4.12 XString

To offer a convenient way to incorporate standard operations on binary data into application logic, the XCO Library provides a standard abstraction for xstrings, `IF_XCO_XSTRING`, that is tightly integrated with the XCO string abstraction `IF_XCO_STRING`.

### Using the Base64 binary-to-text encoding

Built-in conversions are provided to support working with the Base64 binary-to-text encoding. Encoding raw binary data to its Base64 representation and decoding a given Base64 representation to its underlying binary data can be accomplished as follows:

#### ↔ Sample Code

```
" LV_XSTRING can store arbitrary binary data.  
DATA lv_xstring TYPE xstring.  
" LV_BASE64_ENCODING is of type STRING and contains the Base64 encoded version  
" of LV_XSTRING.  
DATA(lv_base64_encoding) = xco_cp=>xstring( lv_xstring  
    )->as_string( xco_cp_binary=>text_encoding->base64  
    )->value.  
" LV_ORIGINAL_XSTRING is of type XSTRING and contains the binary data encoded  
" in LV_BASE64_ENCODING  
DATA(lv_original_xstring) = xco_cp=>string( lv_base64_encoding  
    )->as_xstring( xco_cp_binary=>text_encoding->base64  
    )->value.
```

### Using code pages

An xstring can also be converted to a string by using a code page, e.g. UTF-8:

#### ↔ Sample Code

```
DATA(lv_string) = xco_cp=>xstring( lv_xstring  
    )->as_string( xco_cp_character=>code_page->utf_8  
    )->value.
```

## 4.2.11.17 Metric Providers

The following objects were released for use in the ABAP environment:

Object	Description
CL_GSM_API_TEST	Convenience class to test the metric provider implementation
CX_GSM_API	Metric provider exception class
IF_GSM_API_CONSTANTS	Interface for metric provider constants
IF_GSM_API_METRIC	Interface for metric definition
IF_GSM_API_METRIC_GROUP	Interface for metric group definition
IF_GSM_API_MODEL	Interface for metric provider model definition
IF_GSM_API_PROVIDER	Interface for metric provider definition
IF_GSM_API_TEST	Interface for metric provider testing
IF_GSM_API_TYPES	Interface for metric provider type definitions

These objects are all related to the ABAP repository object called metric provider. You use metric providers as well as these released objects in ABAP Development Tools when you want to create your own metrics for health monitoring of the ABAP system.

## Related Information

[Developing Metrics for Health Monitoring \[page 1189\]](#)

## 4.2.11.18 Integration to SAP BTP Services

[Integrating SAP Document Management Service \[page 2877\]](#)

[Proxy API for the Workflow Capability \[page 1661\]](#)

## 4.2.11.19 Scoping Framework

Learn how to use the scoping API.

Scoping is a mechanism that determines the visibility of certain development object types at configuration time. Depending on your decisions, some object types may be hidden or displayed in the current client.

For instance, in the Business Roles app, not all business catalogs shall be available for the administrator user, because their usage is subject to additional license constraints.

To scope the relevant objects, you can use the API `IF_APS_BC_SCOPE_CHANGE_API`. The interface provides mass-enabled methods for setting the scope state of scoping-enabled TADIR objects in the current client.

#### ⓘ Note

The objects to be scoped must be in the same software component as the caller.

The following TADIR types are supported:

List of TADIR Objects That Can Be Scoped

Constant	TADIR Object
CMHC	Cloud Management Health Check
ESH1	ESH: CDS Search Model
JOB1	Technical Job Definition
PCFN	Predefined Field: Extensible Node
SAJC	Application Job Catalog Entry
SCO1	Communication Scenario
SIA1	Business Catalog
SIA6	IAM App
SIA8	Business Role Template
UIST	Fiori Launchpad Space Template
UIPG	Fiori Launchpad Page Template

To scope one of these objects, you have to provide the object type, object name, and the desired scoping state (ON or OFF).

#### ⓘ Note

Objects that are set to scoping state ON should not be set to OFF. The object must remain available to the customer to guarantee further access to the data maintained via the object.

For further information, see the long text documentation of the API `IF_APS_BC_SCOPE_CHANGE_API`.

## 4.2.11.20 API for Consumption of Business Rules

The purpose of the `CL_FDT_FUNCTION_PROCESS_API` is to enable consumption of BRF+ (Business Rule Framework plus) rules within the ABAP Cloud development environment. It acts as a bridge between your application and BRF+ functions(rules).

The following table shows methods in `CL_FDT_FUNCTION_PROCESS_API` class:

Method	Parameter	Description
Get_Content_List	Importing: IV_FUNCTION_ID  This method is used to get the list of context elements for BRFplus function.	ID of the function to be processed.  The timestamp parameter specifies the data retrieval or function evaluation time. Without a timestamp, the function returns the context list at the current date and time.
	Importing: V_TIMESTAMP:  Importing: IV_TRACE_GENERATION	This Boolean indicates that the trace should be generated with a lean trace or not.
	Exporting: ET_CONTEXT_LIST	The exporting parameter is a table containing the function's context list, which includes the id, name, and data object type of each context.
GET_CONTEXT_VALUE	Importing: IV_FUNCTION_ID  This method is used to get the value of the context object of a BRFplus function in a referenced object.	ID of the function to be processed.  A character-like parameter that specifies the name or ID of the data object involved in the function call.
	Importing: IV_TIMESTAMP  Importing: IV_TRACE_GENERATION	The timestamp parameter specifies the point in time for data retrieval or function evaluation. If no timestamp is provided, the function returns the context value at the current date and time.  This Boolean indicates that the trace should be generated with a lean trace or not.
	Exporting: EV_DATA	This parameter exports the data object's actual value after function execution. Its type "ANY" indicates flexibility to handle any data type, expanding the function's return possibilities.
	Exporting: ER_DATA	This exporting parameter references the data object, enabling its manipulation or examination beyond the function call.
GET_DATA_OBJECT_REFERENCE	Importing: IV_FUNCTION_ID  This method is used to get the data reference of a context object in BRFplus function.	ID of the function to be processed.  Importing: IV_DATA_OBJECT: A character-like parameter that specifies the name or ID of the data object involved in the function call.
	Importing: IV_DATA_OBJECT	A character-like parameter that specifies the name or ID of the data object involved in the function call.

Method	Parameter	Description
	Importing: IV_TIMESTAMP	Importing: IV_TIMESTAMP: The timestamp parameter specifies a specific point in time for data retrieval or function evaluation. If no timestamp is provided, all function context in a referenced object at the current date and time is returned.
	Importing: IV_TRACE_GENERATION	Importing: IV_TRACE_GENERATION: This Boolean indicates that the trace should be generated with a lean trace or not.
	Exporting:  EV_DATA_OBJECT_NAME	This parameter exports the technical name of the data object involved in the operation, providing metadata such as its name or identifier.
	Exporting ER_DATA	This exporting parameter provides a reference to the data object. It does not contain an actual value but rather a pointer to the data object.
MOVE_DATA_TO_DATA_OBJECT	Importing: IR_DATA  This method is used to pass data from external source to BRFplus function's context.	This is a reference to any ABAP data object. It allows the method to directly interact with ABAP data objects provided by the calling program, offering a flexible interface for data processing.
	Importing: IV_FUNCTION_ID	ID of the function to be processed.
	Importing: IV_DATA_OBJECT:	A character-like parameter that specifies the name or ID of the data object involved in the function call.
	Importing: IV_TIMESTAMP	The timestamp parameter is used to define a specific point in time for the data retrieval or function evaluation. If no time stamp is provided, then it takes current date and time.
	Importing: IV_TRACE_GENERATION	This Boolean indicates that the trace should be generated with a lean trace or not.
	Importing: IV_HAS_DDIC_BINDING	Indicates whether the data passed to the method confirms to a DDIC (Data Dictionary) structure as defined in BRF+.
	Exporting: ER_DATA	This exporting parameter exports a reference to a BRFplus data object. This allows the method to return a data object that can be further manipulated or evaluated outside the method.
PROCESS	Importing: IV_FUNCTION_ID	ID of the function to be processed.

Method	Parameter	Description
With this method, a BRFplus function can be processed. You can specify a timestamp for the function to be processed so you can simulate the system behavior as it would have been at that point. Also, you can decide whether the trace for the function execution shall be saved in database for later reference or not.	Importing: IV_TIMESTAMP	The timestamp parameter is used to define a specific point in time for the data retrieval or function evaluation. If no time stamp is provided, the result of the function at the current date and time is returned.
	Importing: IV_TRACE_MODE	Indicates the modes of trace. By default, mode is 'L' (Lean Trace).
	Importing: : IV_SAVE_TRACE	Indicates whether the trace generated for current processing of the Function should be saved in database.
	Importing: IT_ID_VALUE	Table of ID/value pairs (ID/reference to value for each context element). From a performance perspective, this is the most efficient way of triggering the processing of BRFplus. Also, using this parameter requires that you know the IDs of the context elements that you want to pass.
	Exporting: EA_RESULT	Exporting: EA_RESULT:  A variable of a type that is compatible with the result data object of the BRFplus function. You can get an ABAP data object of this type (a reference) by calling method  CL_FDT_FUNCTION_PROCESS_AP I ->IF_FDT_FUNCTION_PROCESS_ API~GET_DATA_OBJECT_REFERE NCE.

## 4.2.11.21 Business Event Logging

This section describes all the released objects in Business Event Logging.

### 4.2.11.21.1 Log Custom Business Events

You can create your own business events with SAP-defined or customer-defined objects. For more information on creating events, see [Business Events](#).

## Define Events for Business Event Logging

The event binding can contain an SAP-defined or a customer-defined object and an operation.

Business events can be defined in two ways: You can define events within the behavior definition at the root node level or you can define events at the respective node level. If you have defined the events at the root node level, ensure that the following annotations are available in the customer event (abstract event entity):

```
@ObjectModel.sapObjectType.name Annotation
```

`@ObjectModel.sapObjectType.name Annotation`: The name of the object components the event refers to.

This can be an SAP-defined object component like SalesOrderItem, or a customer defined object component. For example: Z\_MyObjectItem. If the event refers to the object as a whole or the header of the object, then the object component name should be the same as the object component name in the event binding.

### ↔ Sample Code

```
@ObjectModel.sapObjectType.name: 'Z_MySalesOrderItem'  
define abstract entity ZD_SALESORDERITEMCREATED  
{ ItemNo : item_no  
  Status : Status }
```

```
@Event.sapObjectTypeKey Annotations
```

`@Event.sapObjectTypeKey Annotations`: The key of the object node the event refers to needs to be defined if this is a customer defined object component, or if the key is deviating from SAP standard. The elements of the key must either be defined in the underlying behaviour definition, or the abstract entity itself.

### ↔ Sample Code

```
@ObjectModel.sapObjectType.name: 'Z_MySalesOrderItem'  
@Event.sapObjectTypeKey : [ { element : 'SalesOrder' } , { element :  
  'ItemNo' } ]  
define abstract entity ZD_SALESORDERITEMCREATED  
{ ItemNo : item_no  
  Status : Status }
```

## 4.2.11.21.2 Direct Logging API

You can use the **Direct Logging API** to write to the Business Event Log without creating a Business Event. For an entry in the Business Event Log, the attribute 'source of data' will indicate whether the entry was created via Business Event or via the Direct Logging API.

## Creating an instance to log business event

This class `CL_BEL_DIRECT_LOGGING` offers the functionality to log business events directly into the Business Event Logging component. The class contains the following list of methods to log business events:

Method	Description
<code>IF_BEL_DIRECT_LOGGING~LOG_BUSINESS_EVENT</code>	Logs business events
<code>GET_INSTANCE</code>	Creates an object instance

### Create the instance of the API

The ABAP Class `CL_BEL_DIRECT_LOGGING` can be consumed via the interface `IF_BEL_DIRECT_LOGGING`. Create the instance of the API using `GET_INSTANCE` method of class `CL_BEL_DIRECT_LOGGING`. The method returns the object reference of type `IF_BEL_DIRECT_LOGGING`.

```
8
9  DATA : lo_log_api TYPE REF TO  if_bel_direct_logging.
10
11 lo_log_api = cl_bel_direct_logging=>get_instance( ).
```

### Parameters

The API is invoked via method `LOG_BUSINESS_EVENT` of interface `IF_BEL_DIRECT_LOGGING`. The method takes input as different events and returns the results in terms of return code and an internal table containing processing results of each event.

The method has the following parameters:

- `IT_EVENTS`: This is an input parameter of type table and contains the list of events that need to be logged into the Business Event Logging component. The main components of this parameter are:
  - `EVENT_GUID`: Unique identifier for a given event. This has to be provided by the consumer application. Although it is an optional parameter, we recommend that you provide it. It identifies whether an event has already been logged. By default, a GUID is generated.
  - `SOURCE_SYSTEM`: This identifies the system where the event was generated. It is an optional parameter. By default it would be filled with the current logical system.
  - `TRANSACTION_ID`: An ID uniquely identifying the transaction in which the business event was run. Although it is an optional parameter, we recommend that you provide it. Business events with the same ID are considered to have been run at the same time. By default the transaction\_ID would be filled with the ID of the current kernel transaction.
  - `EXEC_USER`: This identifies the user who performed the business activity that created the event. Although it is an optional parameter, we recommend that you provide it. By default it will be filled with the current logon user.
  - `EXEC_TIMESTAMP`: Timestamp at which the event was created. Although it is an optional parameter, we recommend that you provide it. By default it would be filled with the current timestamp. The timestamp can be in the past.
  - `EVENT_TYPE`: This string carries the metadata such as the object and version information about the event. It is given in a specific format. For example:

### ↔ Sample Code

```
sap.s4.beh.salesorder.v1.SalesOrder.Changed.v1
```

In the event type string, the Object is **SalesOrder**, the Event Operation is **Changed**, and the Event Version is **v1**.

**EVENT\_TYPE** is a mandatory parameter and can be found for a given Business Event in the events section on the SAP API Business Accelerator Hub.

- **EVENT\_DATA:** This is a mandatory parameter. It carries event information such as payload of an event containing the keys and business data. The structure will consist of key fields of the object and object components and the fields of event abstract entity.

For example, in the **sales order changed** event, the **EVENT\_DATA** has structure with "SalesOrder" as the key field and the following fields:

- Object = SalesOrder
- Event Operation = Changed

Using the object details and event operation, determine the Event binding from **EVTB\_PRODUCER** table. Then using the Event Binding, get the CDS behaviour definition from table **EVTB\_EVENT**.

- **EV RETCODE:** The return code may consist of three different values indicating the result of processing.
  - 0 = Successful.
  - 1 = Successful with warnings.
  - 2 = Errors.
- **CT\_RETURN:** This table parameter consists of result of processing of each event which was passed via **IT\_EVENTS** parameter. The components of return table are as
  - **EVENT\_GUID:** Event identifier passed by the consumer application.
  - **EVENT\_TYPE:** Event type passed by the consumer application.
  - **EXEC\_TIMESTAMP:** Event timestamp passed by consumer application.
  - **EVENT\_DATA:** Event specific data passed by consumer application.
  - **TYPE:** Message type: S Success, E Error, W Warning
  - **ID:** Message Class
  - **NUMBER:** Message Number:
    - **MESSAGE\_v1:** Event identifier passed by the consumer application.
    - **MESSAGE\_v2:** Event type passed by the consumer application.
    - **MESSAGE\_v3:** First 50 characters of event data.
    - **MESSAGE\_v4:** Event timestamp passed by consumer application.

### Troubleshooting issues

The API can throw a Warning(W) return code in case the event is already existing in BEL database or event is not activated to be logged into BEL framework. The API can return error(E) return code in case the mandatory parameters are not passed or passed with initial values, if authorization check fails , if the **EVENT\_DATA** structure is passed incorrectly.The details of the error/warning can be seen in **CT\_RETURN** table parameter of the API. Some examples of error/warning are given below:

- Business Event is not defined (E) : The supplied business even is not defined either in Business Event Handling or RAP framework.
- No change auth for Object (E) : No authorization to log the business events directly.

- Business event with Identical ID exists (W) : The business event supplied already exists in system.
- Business Event not enabled for logging (W) : The business event is not enabled for logging in the system.
- Business event processed successfully (S): Business event is successfully logged into the system.
- Business event data not supplied (E): Event related data is not supplied.
- Business event keys can't be determined (E): For the given event, Business Object keys couldn't be determined.
- Keys can't be retrieved from event data (E): Business object keys are not completely specified in event data.

### 4.2.11.21.3 Accessing Business Event Log Data using CDS Views

The following views are released for key user extensibility:

CDS Views	Description
C_BUSEVTLOGEVENTDEX_2	This CDS view provides business event header data.
C_BUSEVTLOGPAYLOADDEX_2	This CDS view provides a subset of event payload details.
C_BUEVLGEVTFULLPAYLOADJSONDEX	This CDS view provides full business event data (payload) in JSON format.

## 4.2.12 UI Development

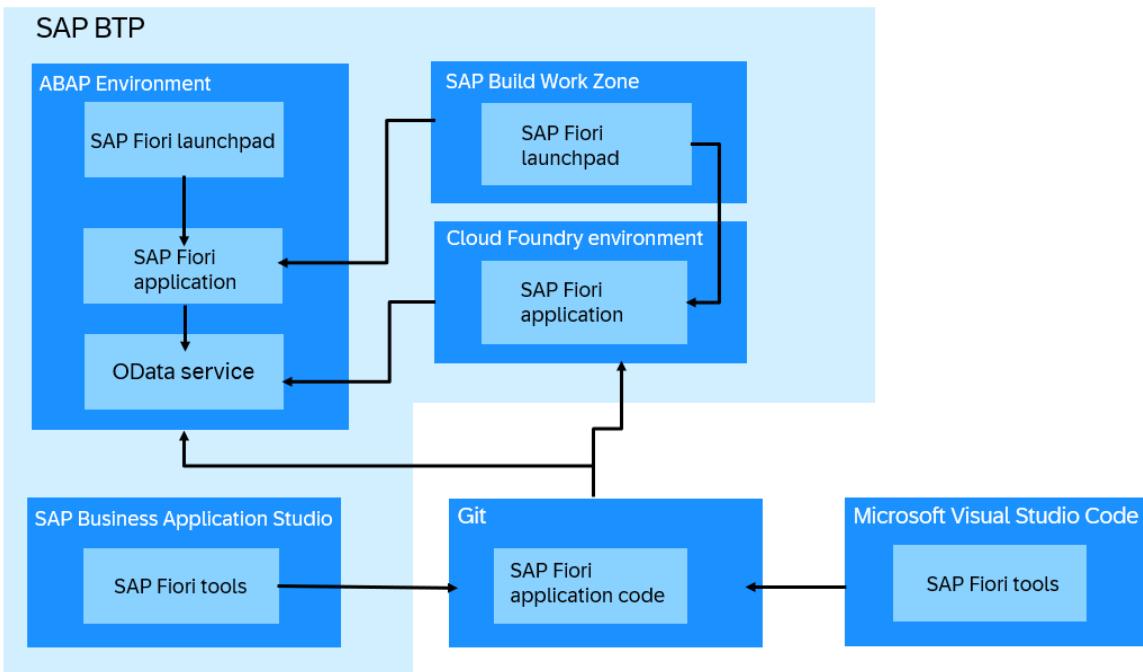
Learn more about the different options to develop SAP Fiori applications for an OData service, where to store the application, and how to enable business users to access the application from SAP Fiori launchpad.

### Overview

You can develop SAP Fiori applications for OData services that are exposed as a UI service. See [OData Service Consumption](#).

- SAP Fiori is a design system that enables you to create business apps with a consumer-grade user experience, turning casual users into SAP experts with simple screens that run on any device. Apps that follow the Fiori design principle can be built using a combination of SAP UI5 and SAP Fiori Elements. See [SAP Fiori](#).
- SAPUI5 is a development framework that a developer would use to actually build a front-end application that follows the Fiori design guidelines. UI5 apps are built using HTML5, JS, XML, OData and JSON, and are based on the Model, View, Controller (or MVC) design pattern. See [SAPUI5](#).
- SAP Fiori Elements is a framework that comprises the most commonly used floorplan templates and is designed to speed up development by reducing the amount of frontend code needed to build SAP Fiori

apps, and driving UX consistency and compliance with the latest SAP Fiori design guidelines. See [SAP Fiori Elements](#).



- <https://help.sap.com/docs/btp/sap-fiori-launchpad-for-sap-btp-abap-environment/sap-fiori-launchpad-user-guide?version=Cloud> [https://help.sap.com/docs/btp/sap-fiori-launchpad-for-sap-btp-abap-environment/sap-fiori-launchpad-user-guide?version=Cloud]
- <https://help.sap.com/docs/btp/sap-business-technology-platform/sap-fiori-applications-in-cloud-foundry-environment?version=Cloud> [https://help.sap.com/docs/btp/sap-business-technology-platform/sap-fiori-applications-in-cloud-foundry-environment?version=Cloud]
- <https://help.sap.com/docs/btp/sap-business-technology-platform/sap-business-application-studio?version=Cloud> [https://help.sap.com/docs/btp/sap-business-technology-platform/sap-business-application-studio?version=Cloud]
- <https://www.git-scm.com/> [https://www.git-scm.com/]
- [https://help.sap.com/docs/SAP\\_FIORI\\_tools/17d50220bcd848aa854c9c182d65b699/f09752ebcf63473e9194ea29ca232e56.html?q=sap%20fiori%20tools](https://help.sap.com/docs/SAP_FIORI_tools/17d50220bcd848aa854c9c182d65b699/f09752ebcf63473e9194ea29ca232e56.html?q=sap%20fiori%20tools) [https://help.sap.com/docs/SAP\_FIORI\_tools/17d50220bcd848aa854c9c182d65b699/f09752ebcf63473e9194ea29ca232e56.html?q=sap%20fiori%20tools]
- [https://help.sap.com/docs/SAP\\_FIORI\\_tools/17d50220bcd848aa854c9c182d65b699/17efa217f7f34a9eba53d7b209ca4280.html?q=cloud%20foundry%20environment](https://help.sap.com/docs/SAP_FIORI_tools/17d50220bcd848aa854c9c182d65b699/17efa217f7f34a9eba53d7b209ca4280.html?q=cloud%20foundry%20environment) [https://help.sap.com/docs/SAP\_FIORI\_tools/17d50220bcd848aa854c9c182d65b699/17efa217f7f34a9eba53d7b209ca4280.html?q=cloud%20foundry%20environment]

In order to develop SAP Fiori applications, you use SAP Business Application Studio with SAP Fiori tools extensions. SAP Business Application Studio supports integration with Git so that you can use Git as source control system and store the code of SAP Fiori applications in remote git repositories. Once you have

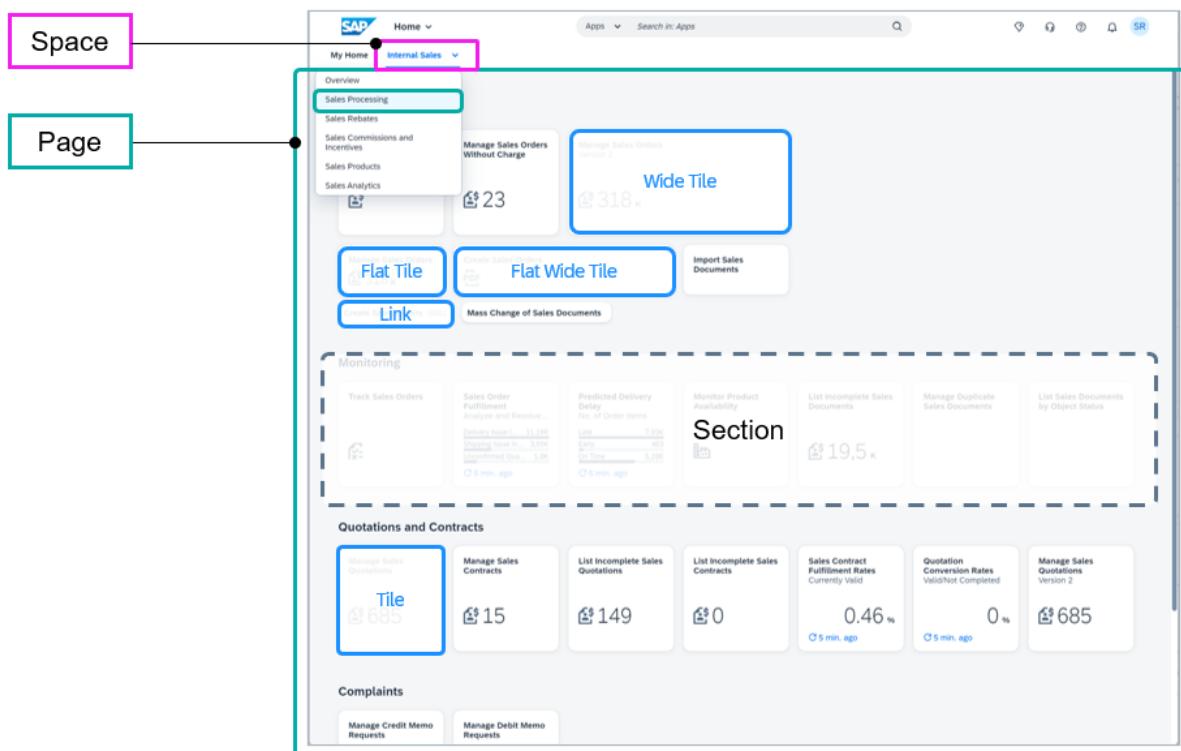
implemented the custom SAP Fiori application, you can deploy it to the ABAP environment or the Cloud Foundry environment. You can launch the deployed apps standalone or embed them into the SAP Fiori launchpad. The ABAP environment comes with an embedded Fiori Launchpad and provides access to Fiori apps that are stored on the SAPUI5 ABAP repository. Apps stored in the HTML5 application repository can be added to a launchpad provided by the SAP Build Work Zone. It is also possible to add apps stored on the SAPUI5 ABAP repository to launchpads provided by the SAP Build Work Zone. See [Integration Scenario](#)"

Spaces and pages structure the way apps and other content appears in the SAP Fiori launchpad. They offer more flexibility to influence the launchpad layout for different user groups. Each user can see one or more spaces that contain one or more pages. The pages show apps clustered in different sections.

A **Space** is assigned to a business role and offers a structured layout of the business role content. Each space must consist of at least one page.

A **Page** holds the actual layout information for the corresponding space. It groups the apps visualized by tiles into different sections from the work perspective of the end user.

The following graphic illustrates the concept of spaces and pages.



Read [Managing Launchpad Spaces and Pages](#) for an in-depth description of the concept in the SAP Business Technology Platform. For more information on how to use spaces and pages in SAP Build Work Zone, please refer to [Manual Configuration of Spaces and Pages](#).

## SAP Fiori Applications in the ABAP environment

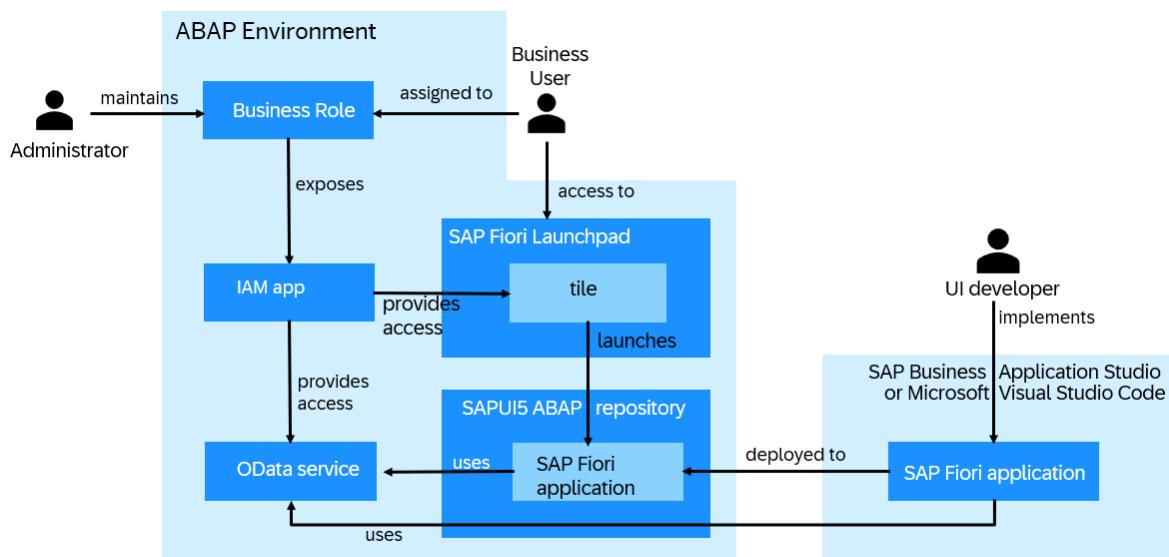
If you want to deploy an SAP Fiori application to the ABAP environment, the following users are involved:

- A UI developer implements the SAP Fiori application against the OData service and defines the tile as part of the application. Once this is done, the developer deploys the SAP Fiori application to the SAPUI5 ABAP repository.

### ⓘ Note

The SAPUI5 ABAP repository is part of the ABAP environment and is the umbrella term for the single SAPUI5 repository of each application. Technically, the SAPUI5 ABAP repository is based on the Business Server Page (BSP) repository. Each SAPUI5 repository is represented by an individual BSP application. The SAPUI5 ABAP repository is also used for delivering SAPUI5 apps of the ABAP environment.

- An administrator in the ABAP environment provides access to the OData service and the tile via a business role.
- A business user that is assigned to the business role can access the tile from SAP Fiori launchpad and launch the application.



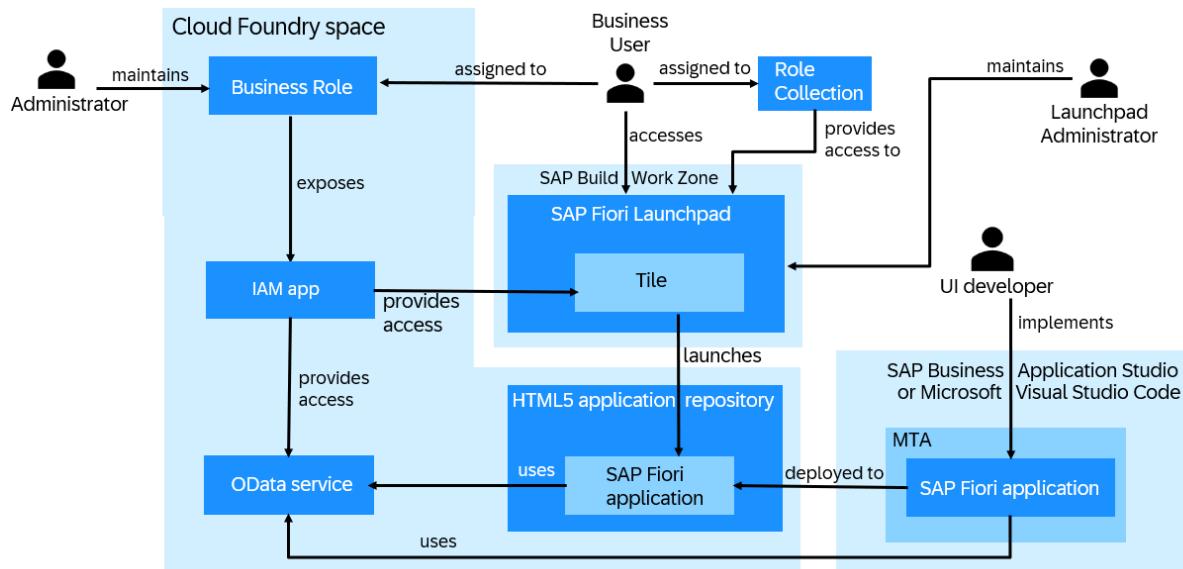
## SAP Fiori Application in the Cloud Foundry environment

If you want to deploy an SAP Fiori application to the Cloud Foundry environment, the following users are involved:

- A UI developer implements the SAP Fiori application against the OData service as part of a multi-target application (MTA).  
An MTA is logically a single application comprised of multiple parts created with different technologies, which share the same lifecycle. The developers of the MTA describe the desired result using the MTA model, which contains MTA modules, MTA resources, and interdependencies between them.
- The developer defines the tile as part of the application. Once this is done, the MTA is deployed to the Cloud Foundry space. By doing so, the SAP Fiori application of the MTA is deployed to the HTML5 application repository. The developer can deploy the MTA either manually or as part of a CI/CD pipeline. See [Continuous Integration and Delivery \(CI/CD\)](#).

The HTML5 application repository is an SAP BTP service that enables central storage of HTML5 applications' static content on the SAP BTP, Cloud Foundry environment. See [HTML5 Application Repository](#).

- An administrator in the ABAP environment provides access to the OData service via a business role. A launchpad administrator enables access to the tile via a role collection.
- A business user that is assigned to the business role and role collection can access the tile from SAP Fiori launchpad and launch the application.



## Related Information

[SAP Business Application Studio](#)

[SAP Fiori Tools](#)

[SAP Fiori Overview](#)

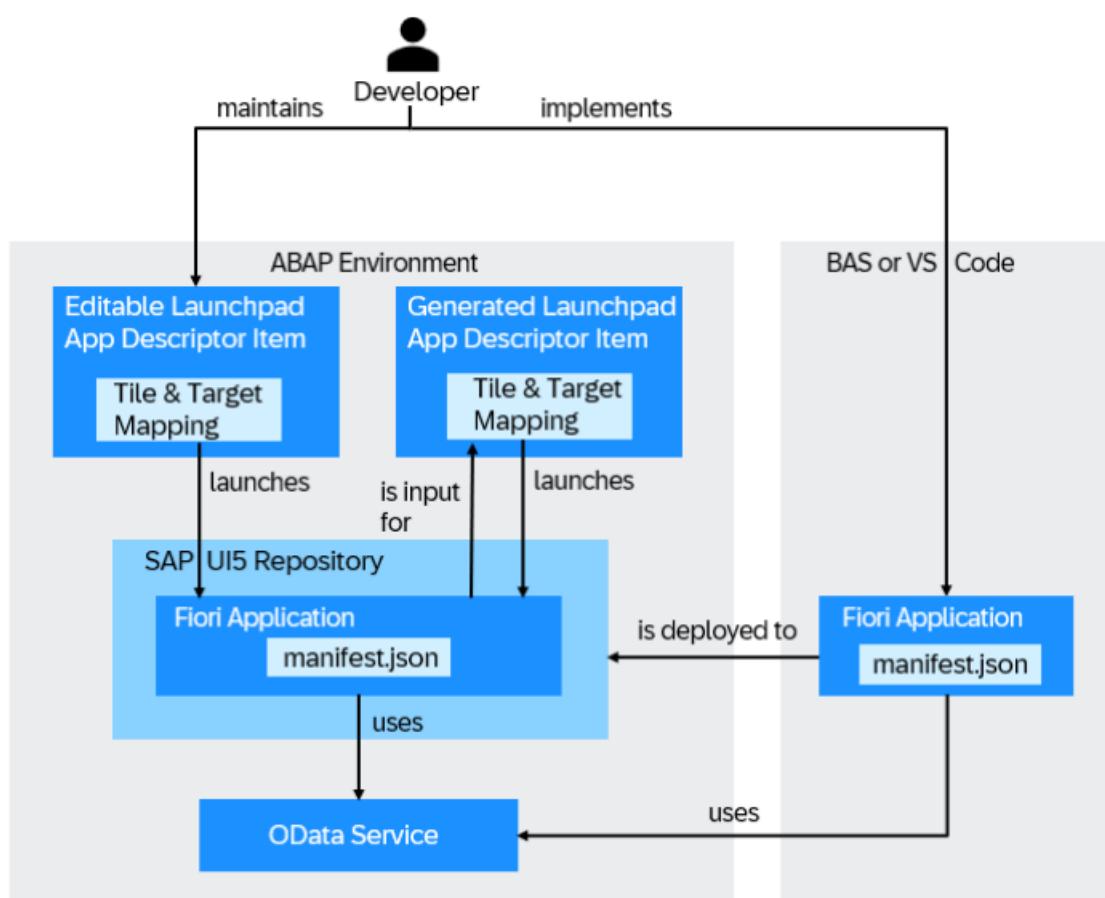
[Visual Studio Code](#)

[Test Automation](#)

## 4.2.12.1 SAP Fiori Applications in the ABAP Environment

When you plan to create an SAP Fiori application in the ABAP environment and want to provide this application to business users, you may want to get a better understanding of the required steps before getting started with the UI development.

### Implementing and Deploying the UI



### Implementing an SAP Fiori Application

To implement an SAP Fiori application for an OData service, you can use the Fiori tools application generator in SAP Business Application Studio or Visual Studio Code. When generating the application, you can select one of the available OData services for UI development and choose the main entity. Amongst others, the generator will create the `manifest.json` file. The manifest defines static information about the application, such as the name of the application or OData service used. You can add a SAP Fiori launchpad configuration to the `manifest.json` file, for example, a tile with an icon. The SAP Fiori launchpad configuration can be added from the application generator or from the application information page.

After generating the app, you can run the application in preview mode.

### **⚠ Restriction**

- Only OData services from your custom service binding are supported.
- API services on SAP Business Accelerator Hub are not supported. They are released for external consumption only.

To discover the available OData services, the `SAP_A4C_BC_DEV_UID_PC` business catalog needs to be assigned. In order to run the application in preview mode, the `SAP_CORE_BC_EXT_TST` business catalog has to be assigned. This business catalog is contained in the `SAP_BR_DEVELOPER` business role template.

## **Deploying an SAP Fiori Application**

To deploy the SAP Fiori application, you have to create a deployment configuration and provide the target package and workbench request. You can create the deployment configuration from the application generator or from the application information page.

When deploying the application to the SAPUI5 ABAP repository in the ABAP environment, the following objects are automatically created and recorded on the workbench request.

- A business server page (BSP) application, which stores the code of the SAP Fiori application
- A folder in the MIME repository, which stores all MIME objects, such as graphics or icons of the app
- A node in the Internet Communication Framework (ICF), which enables clients (browser) to access the code stored in the BSP application
- SAP Fiori launchpad app descriptor items which enable launching the app from the Fiori launchpad

You can find the BSP application and the generated launchpad app descriptor items in your deployment package in ABAP development tools for Eclipse. The folder in the MIME repository and the node in the ICF are not visible in ABAP development tools for Eclipse but are listed in the transport request. Alternatively, you can use CDS view `I_CustABAPObjDirectoryEntry` to inspect those objects in the ABAP object directory. The CDS view gives you customer object centric visibility of the classic ABAP table `TADIR`.

The launchpad app descriptor items are created from the SAP Fiori launchpad configuration in the `manifest.json` file. For more information, please navigate to the entry [crossnavigation](#) in table `sap.app` of [Descriptor for Applications, Components, and Libraries \(manifest.json\)](#). Each app descriptor item consists of a target mapping and a tile. The target mapping maps the deployed app (navigation target) to the intent (combination of semantic object and action). Moreover, the generated Fiori launchpad app descriptor items follow the application deployment lifecycle and cannot be edited in ABAP development tools for Eclipse.

### **ⓘ Note**

If you change the ID of the SAP Fiori launchpad configuration intent in the `manifest.json` file of the Fiori application and deploy the latter, existing references from the SAP Fiori launchpad to the corresponding Fiori launchpad app descriptor item will break.

The Fiori UI is deployed in ABAP language version ABAP for Cloud development which restricts access to data sources. Data sources must be released for system-internal use in cloud development or must be in the same software component. For more information, see [Use System-Internally \(C1\)](#).

To deploy the application, the `SAP_A4C_BC_DEV_UID_PC` business catalog needs to be assigned. This business catalog is contained in the `SAP_BR_DEVELOPER` business role template.

## Creating Launchpad App Descriptor Items

In addition to the launchpad app descriptor items that are generated automatically from the `manifest.json` file during deployment, you can also create launchpad app descriptor items manually in ABAP development tools for Eclipse. The launchpad app descriptor item consists of a target mapping and one or more tiles. The target mapping maps the application (navigation target) to the intent (combination of semantic object and action).

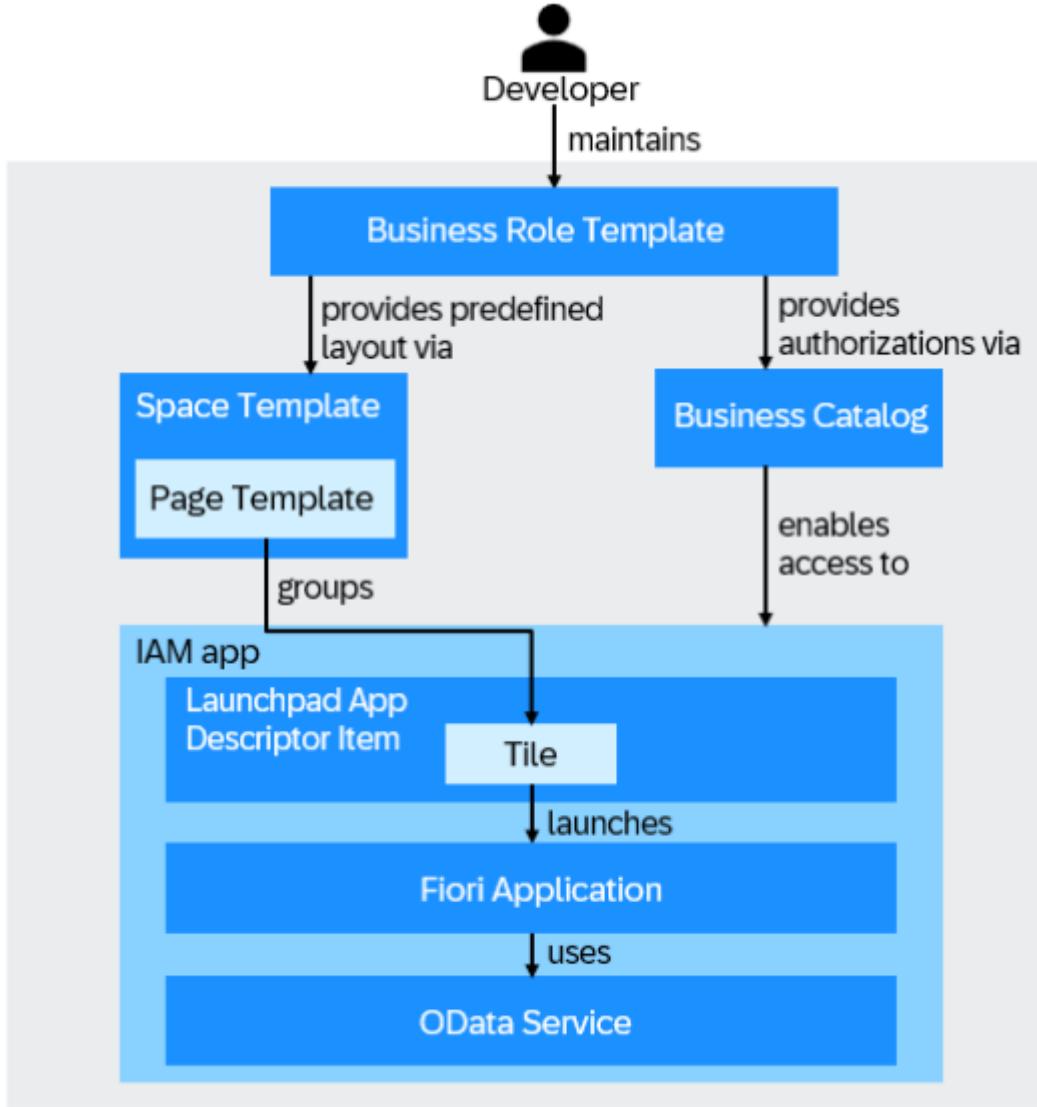
If you want to provide different tiles for the app, for example, a static tile and a dynamic tile with a counter, then you can add additional tiles to the launchpad app descriptor. In this case, users can personalize the launchpad with the tile of their choice. If you are intending to provide different tiles for different personas, then it is recommended to create distinct launchpad app descriptor items. For more information, see [Working with Launchpad App Descriptor Items](#). The properties provided under [Editing Launchpad App Descriptor Items](#) can be maintained.

These manually created launchpad app descriptor items can be created with a freely chosen name. Keep in mind that the lifecycle of those manually created objects is decoupled from the Fiori application deployment lifecycle, so they can be edited in ABAP development tools for Eclipse independently.

It is recommended to utilize launchpad app descriptor items created from the `manifest.json` file during app deployment. This read-only app-descriptor item serves as reference and fallback tile for any other tiles that are created in addition to that. You can manually create these items in ABAP development tools for Eclipse if specialized tiles are needed for an already deployed Fiori application. For example, if you wish to create a separate tile that displays a specific subset of data and has a different title, you can manually create an additional app descriptor item with a dedicated tile. This additional app descriptor item can also be used for enabling access and defining the predefined launchpad layout.

## Enabling Access and Providing Predefined Launchpad Layout

After you have deployed the SAP Fiori application and created Fiori launchpad app descriptor items, either automatically from the manifest or manually in ABAP development tools for Eclipse, you need to create an IAM app and assign the created SAP Fiori launchpad app descriptor item to the IAM app. Furthermore, you need to create a business catalog and assign the IAM app to the business catalog. The business catalog enables access to the SAP Fiori application.



Additionally, you can provide a predefined launchpad layout by creating a space template and a page template referring to the tile of the SAP Fiori launchpad app descriptor item.

Moreover, you can maintain a business role template to have a streamlined process for an administrator to set up business roles by providing predefined roles with predefined layouts.

This saves time and effort for an administrator but still allows for a more tailored user experience.

While the business catalog defines the access to the apps, the space and page template determine which apps are visible for the users on the Fiori Launchpad. The business role template combines the business catalogs and space templates that are typically needed for such a role.

### ⚠ Restriction

Creation of Space Templates and Page Templates

Space templates can't be merged with other space templates directly. To merge a space template with another space template, one of the templates must be copied to create a space first.

When you want to merge one or more customer-created spaces with a predefined space (e.g. delivered by SAP) or a template, open the general details for the customer-created space, then enter the space ID of the predefined space or template in the field *Merge ID*. The same is true for the merging of page templates.

## Scoping

Business catalogs, business role templates as well as Fiori space and page templates are not visible in the development system by default and therefore cannot be used.

To use space and page templates in the development system, you need to publish it locally. After the business catalog and role template are published, they are visible in the Business Catalogs and Business Role Templates apps. To use the space and page templates in the development system, you need to programmatically scope them. See [Scoping Space and Page Templates \[page 1852\]](#) for more details about scoping. After the page and pace templates are scoped, they are visible as predefined spaces and pages in the [Manage Launchpad Spaces](#) and [Manage Launchpad Pages](#) apps. In non-development systems the space and page templates are scoped automatically during import into the system.

## Preview

As a developer you can launch the preview of the page template in the [Manage Launchpad Pages](#) app from the ABAP development tools for Eclipse editor. To preview the created page template, you need authorization for the [Manage Launchpad Pages](#) app.

## Translation

### *Maintain Translations*

When it comes to managing translation, there are basically two areas: Texts from the ABAP development objects and texts from the SAP Fiori application, e.g., texts for the IAM app or IAM business catalog. Texts from the SAP Fiori application are usually defined in i18n (internationalization) property files, e.g. the texts for the tile used in the `manifest.json`.

To handle texts from the ABAP development objects, you use the [Maintain Translations App](#). This application provides a user-friendly interface for managing translations within the ABAP environment. It allows you to easily maintain and update translations used in ABAP development objects.

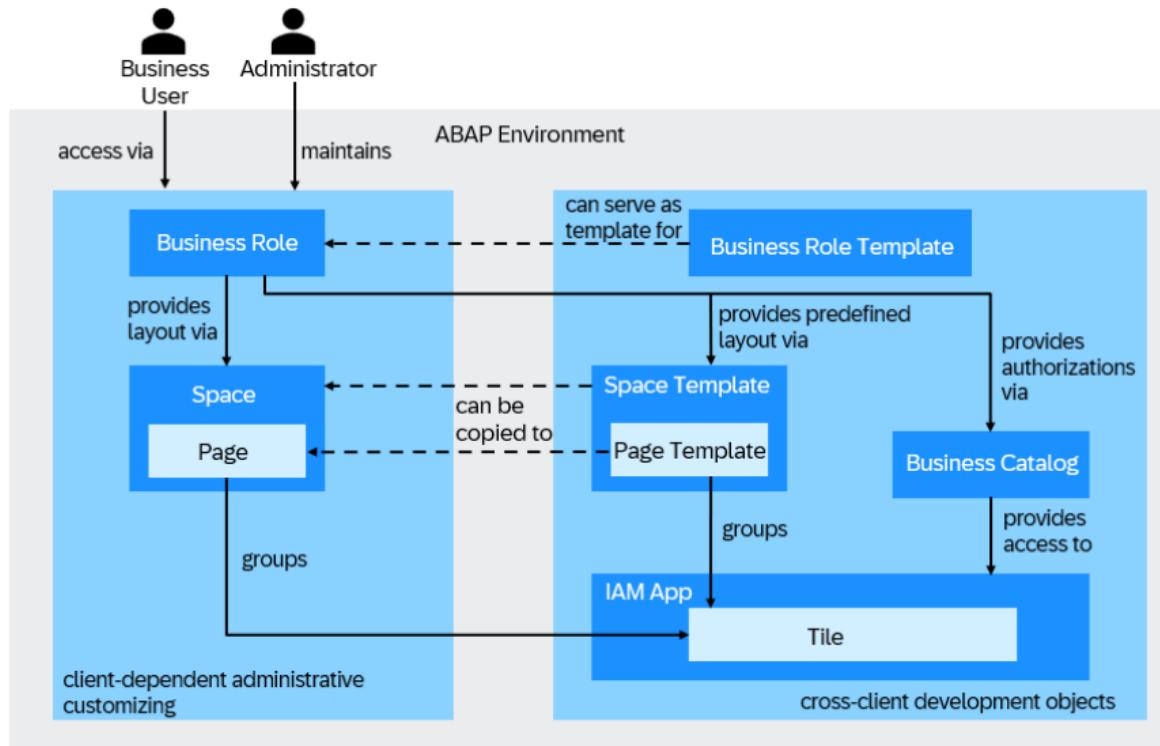
For non-ABAP based texts, such as those found in the SAP Fiori application, a different approach is needed. In this case, you need to follow the Fiori translation process. Please refer to [Translate Texts in SAP Fiori](#) for more details.

### ⓘ Note

Texts of launchpad app descriptor items created manually in ABAP development tools for Eclipse can be translated in the [Maintain Translations](#) app. Texts of launchpad app descriptor items generated from the `manifest.json` file during application deployment origin from i18n property files and follow the translation approach for non-ABAP based texts.

## Providing Access and Launchpad Layout

You as an administrator create a **business role**, so that the application appears in the SAP Fiori launchpad for **business users** that are assigned to the business role. Using role templates as well as predefined space and page templates simplifies the process for an administrator of setting up business roles.



To make an SAP Fiori application accessible on the SAP Fiori launchpad home page of business users, you must perform the following steps:

- Create a business role and assign at least one **business catalog** to the role. The business role controls the access to your applications. The business catalog contains the actual authorizations that allow users to access apps.
- You can either assign a predefined space (space template) to the **business role**, copy an existing one, or create a new launchpad space and page and then add apps to the page. The space and page control the layout of the Fiori launchpad. More information and tips on creating and handling spaces and pages can be found in [Best Practices for Managing Spaces and Pages](#).
- You can also create a business role from a business role template and adapt the business role to your needs.
- Business users that are assigned to the business role can now access the app.
- You can also create spaces, pages, and business roles for testing purposes locally in the development system. If you want to import these artifacts into the production system, you must transport the spaces, pages, and business roles into the production system via a customizing request. See [Export Customizing Transports](#) for more details on how to use it.
- If you are using a group-based layout in the SAP Fiori launchpad, you must enable SAP Fiori launchpad spaces in the [Manage Launchpad Settings](#) app.

 Note

This is a one-time effort.

## Related Information

[Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio \[page 1841\]](#)

### 4.2.12.1.1 Develop an SAP Fiori Application UI and Deploy it to ABAP Using SAP Business Application Studio

Get an overview about how to create and deploy an SAP Fiori application to ABAP using SAP Business Application Studio.

If you need further assistance with creating and deploying an application, check out the tutorial [Create an SAP Fiori App and Deploy it to SAP BTP, ABAP Environment](#).

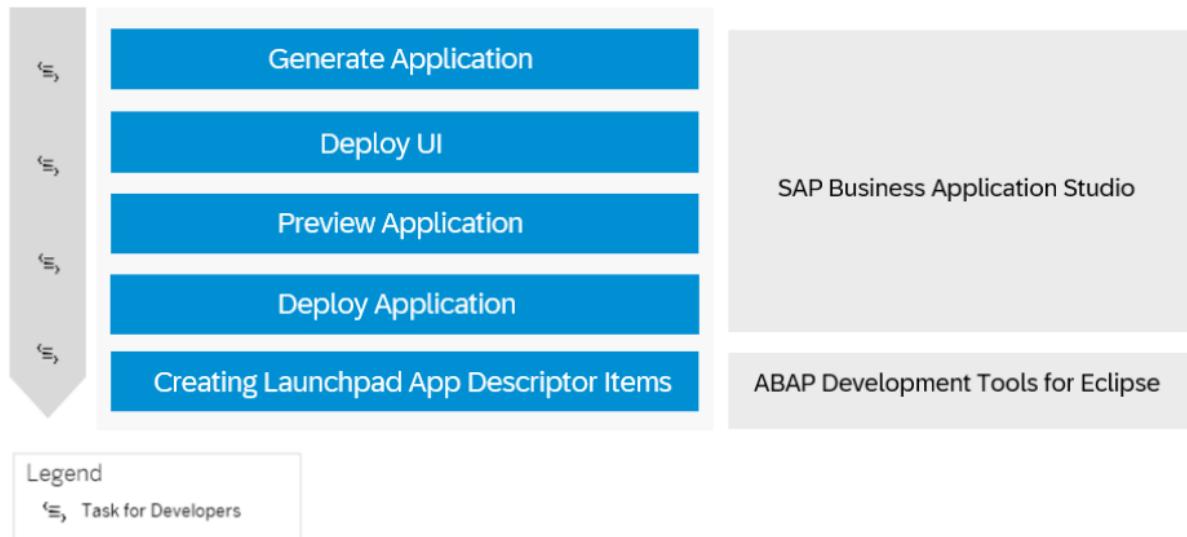
## Prerequisites

- You have set up SAP Business Application Studio. See [Setup of UI Development in SAP Business Application Studio](#).
- You have established trust by setting up a custom Identity service. See [Setup of a Custom Identity Service](#).

## 1. Generating and Deploying Your Application

 Prerequisites

- You have created a development package in ABAP Development Tools for Eclipse. See [Creating ABAP Packages](#).
- You have exposed a RAP business service as an OData service. See [Using Service Binding Editor for OData V2 Service](#).
- Business catalog `SAP_A4C_BC_DEV_UID_PC` is assigned to your user, which allows you to discover available OData services and deploy your application.
- Business catalog `SAP_CORE_BC_EXT_TST` is assigned to your user, which allows you to preview your application. See [Business Catalog for Key User Tasks](#).
- You have an open transport request.



1. As a developer user in SAP Business Application Studio, generate an SAP Fiori application. See [Generate an Application](#).
    1. In the *Data Source and Service Selection* section of the Template Wizard, select the following values:
      - Data source: *Connect to a System*.
      - System:
        - (Option 1) Select the destination that you have created for the SAP Business Application studio integration (`SAP_Business_Application_Studio`). See [Creating a Destination to the ABAP System for SAP Business Application Studio](#).
        - (Option 2) Select *ABAP Environment on SAP Business Technology Platform*. From the ABAP environment drop-down menu, choose a service instance.
- ⓘ Note**

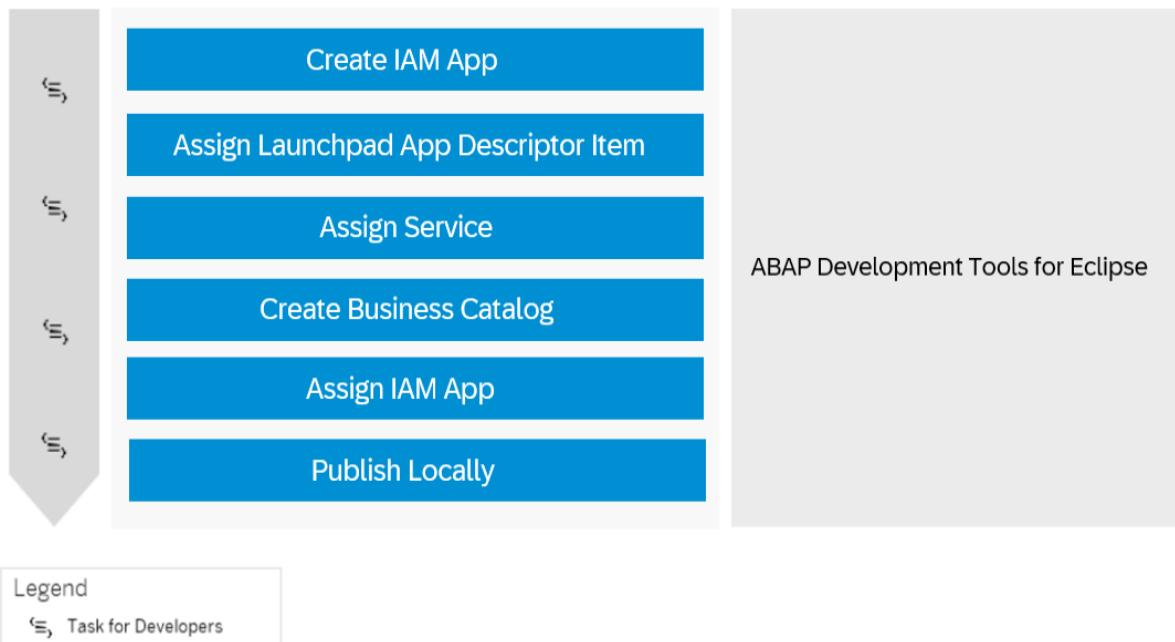
You have to log on to your Cloud Foundry space by executing command `cf login` in the terminal or by navigating to [View > Find Command > CF: Login to Cloud Foundry](#). When you're prompted to enter the API endpoint, org name, and space, you can navigate to your subaccount in the SAP BTP cockpit, where you can find this information.
1. Continue with choosing a service.
  2. Add a deployment configuration. See section [Add deployment configuration > ABAP system](#) in [Additional Configuration](#). If you want to create your deployment configuration later, see [Generate Deployment Configuration ABAP](#).
  3. Add an SAP Fiori launchpad configuration for your UI project. See section [Add FLP configuration](#) in [Additional Configuration](#). If you want to create you FLP configuration later, see [SAP Fiori Launchpad Configuration](#).
  2. Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).
  3. Now you can preview the generated SAP Fiori application. See [Preview an Application](#).
  4. Deploy the SAP Fiori UI by executing command `npm run deploy` in the terminal of your project. See [Deployment to ABAP](#).

4. On top of these launchpad app descriptor items that are created automatically during app deployment, you may also create such objects manually in ABAP development tools for Eclipse. Refer to [Creating Launchpad App Descriptor Items](#).

## 2. Creating and Publishing Your Identity and Access Management (IAM) App

### ⓘ Prerequisite

Business catalog SAP\_A4C\_BC\_DEV\_PC is assigned to your user, which is required for development with ABAP Development Tools. See [Business Catalogs for Development Tasks \[page 2548\]](#).



- To manage access to your SAP Fiori application, you need to log on as a developer in ABAP development tools for Eclipse to create an Identity and Access Management (IAM) application, assign a launchpad app descriptor item for your UI5 application and a service, and maintain authorizations (steps 1-3 in the figure above). See [Creating an IAM App for the Business Service](#).
- Once you have created your IAM app, you have to create a business catalog. See [Creating a Business Catalog](#).
- Assign your IAM app to the business catalog.
- Publish the IAM app and business catalog locally.

## Next Step

Launch your app in SAP Fiori launchpad. See [Add Your App to SAP Fiori Launchpad](#).

## Related Information

[SAP Business Application Studio](#)

[SAP Fiori Tools](#)

[SAP Fiori Overview](#)

[Tutorial: Develop and Run SAP Fiori Application With SAP Business Application Studio](#) 

[Tutorial: Integrate List Report into ABAP Fiori Launchpad](#) 

### 4.2.12.1.2 Develop an SAP Fiori Application UI and Deploy it to ABAP Using Visual Studio Code

Get an overview about how to create and deploy an SAP Fiori application to ABAP using Visual Studio Code.

For further assistance with creating and deploying an application into SAP Fiori launchpad, see [Create a SAP Fiori App in Visual Studio Code and Deploy it to SAP BTP, ABAP Environment](#) .

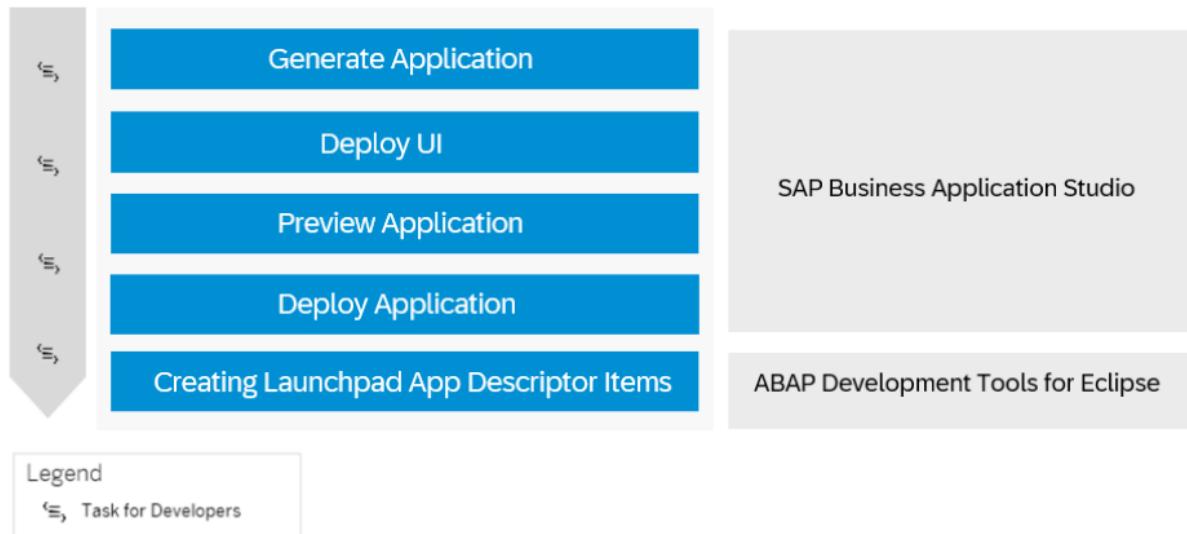
## Prerequisites

- You have installed and set up Visual Studio Code including the SAP Fiori tools extensions. See [Visual Studio Code](#).
- You have access to a RAP business service that has been exposed as an OData service. See [Using Service Binding Editor for OData V2 Service](#).
- To establish a connection with your ABAP environment system, you either have to be a space developer in the ABAP environment instance or have access to a service key in the ABAP environment instance. See [Add Space Members Using the Cockpit](#) and [Creating Service Keys](#) in the ABAP service instance.
- You have established trust by setting up a custom Identity service. See [Setup of a Custom Identity Service](#).

## 1. Generating and Deploying Your Application

### Prerequisites

- You have created a development package in ABAP Development Tools for Eclipse. See [Creating ABAP Packages](#).
- You have exposed a RAP business service as an OData service. See [Using Service Binding Editor for OData V2 Service](#).
- Business catalog `SAP_A4C_BC_DEV_UID_PC` is assigned to your user, which allows you to discover available OData services and deploy your application.
- Business catalog `SAP_CORE_BC_EXT_TST` is assigned to your user, which allows you to preview your application. See [Business Catalog for Key User Tasks](#).
- You have an open transport request.



1. As a developer user in Visual Studio Code, generate an SAP Fiori application. See [Generate an Application](#). In the [Data Source and Service Selection](#) section of the Template Wizard, select the following values:

- Data source: [Connect to a System](#).
- System: [New System](#)
- System type: [ABAP Environment on SAP Business Technology Platform](#)
- ABAP environment definition source:
  - (Option 1) [Discover a Cloud Foundry Service](#)

### ⓘ Note

You have to log on to your Cloud Foundry space by executing command `cf login` in the terminal or by navigating to [View > Find Command > CF: Login to Cloud Foundry](#). When you're prompted to enter the API endpoint, org name, and space, you can navigate to your subaccount in the SAP BTP cockpit, where you can find this information.

- (Option 2) [Upload a Service Key File](#)

- Continue with choosing a system name and service.

In the [Project Attributes](#) section, add deployment configuration and FLP configuration to your UI project. See [Additional Configuration](#).

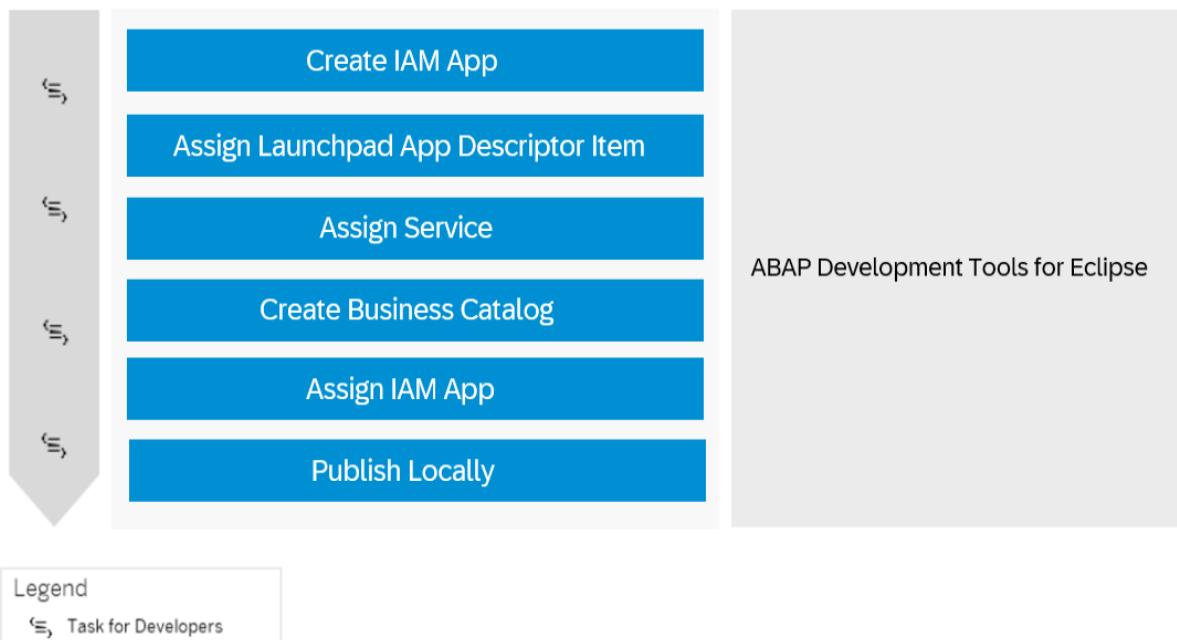
If you want to create your deployment configuration later, see [Generate Deployment Configuration ABAP](#). If you want to create your FLP configuration later, see [SAP Fiori Launchpad Configuration](#).

2. Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).
3. Now you can preview the generated SAP Fiori application. See [Preview an Application](#).
4. Deploy the SAP Fiori UI by executing command `npm run deploy` in the terminal of your project. See [Deployment to ABAP](#).
5. On top of these launchpad app descriptor items that are created automatically during app deployment, you may also create such objects manually in ABAP development tools for Eclipse. Refer to [Creating Launchpad App Descriptor Items](#).

## 2. Creating and Publishing Your Identity and Access Management (IAM) App

### ⓘ Prerequisite

Business catalog SAP\_A4C\_BC\_DEV\_PC is assigned to your user, which is required for development with ABAP Development Tools. See [Business Catalogs for Development Tasks \[page 2548\]](#).



1. To manage access to your SAP Fiori application, you have to log on as a developer in ABAP Development Tools for Eclipse to create an Identity and Access Management (IAM) application, assign a launchpad app descriptor item for your UI5 application and a service, and maintain authorizations (steps 1-3 in the figure above). See [Creating an IAM App for the Business Service](#).
2. Once you have created your IAM app, you have to create a business catalog. See [Creating a Business Catalog](#).
3. Assign your IAM app to the business catalog.
4. Publish the IAM app and business catalog locally.

### Next Step

Launch your app in SAP Fiori launchpad. See [Add Your App to SAP Fiori Launchpad \[page 1851\]](#).

### Related Information

[Visual Studio Code](#)

[SAP Fiori Tools](#)

SAP Fiori Overview

Tutorial: Integrate List Report into ABAP Fiori Launchpad 

Tutorial: Create an SAP Fiori App in Visual Studio Code and Deploy it to SAP BTP, ABAP Environment 

### 4.2.12.1.3 Extend an SAP Fiori App and Deploy it to ABAP Using SAP BAS

Learn how to extend an SAP Fiori application with an application variant by creating an SAPUI5 adaptation project and deploying it to ABAP using SAP Business Application Studio.

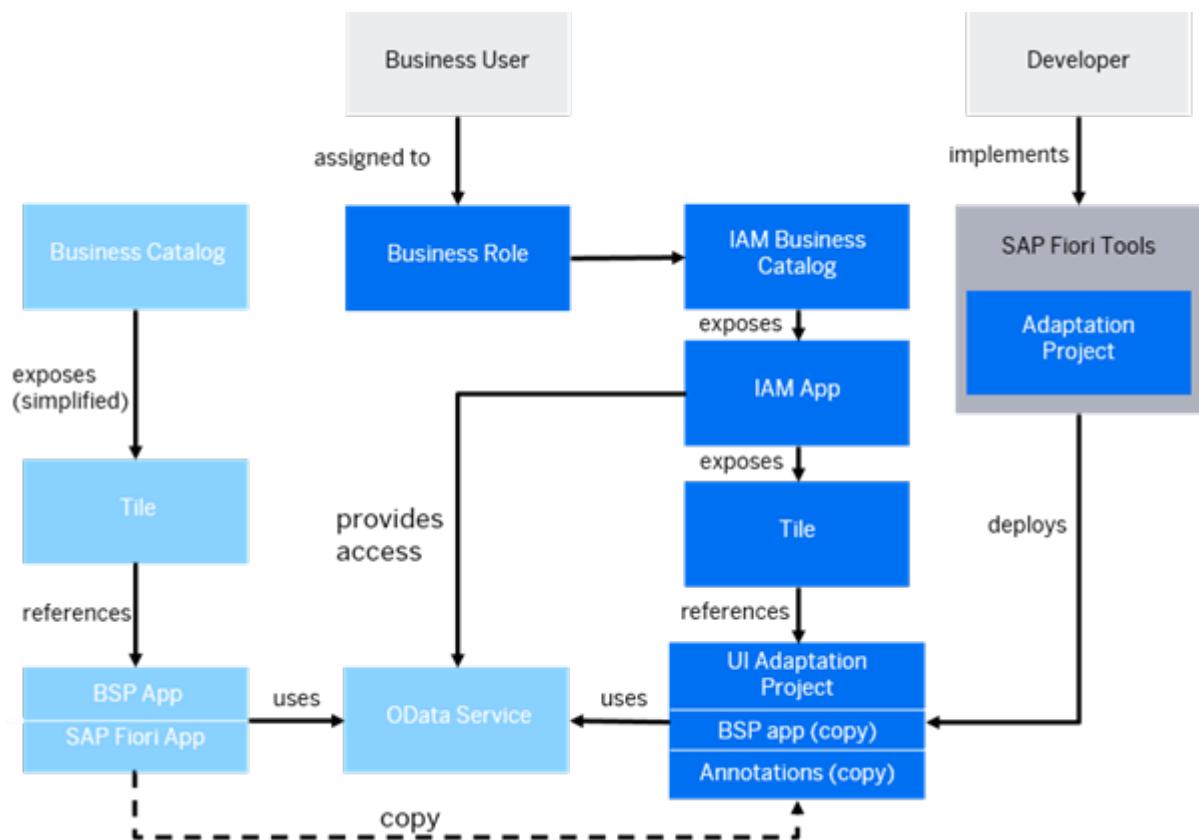
#### Overview

If an SAP Fiori application doesn't fully support your business process or the specific needs of some of your business users, and if the SAP Fiori application is released for extensibility, you can extend the SAP Fiori application with an application variant while the original SAP Fiori application remains available and unchanged. In an application variant, you can extend the original code to define changes that fulfill the specific business requirements of different business roles, user groups, organizational units, and countries. For example, you might want to create an application variant to simplify and streamline a specific process by hiding fields not required by casual users, visualizing data as a chart, or changing a responsive table to a grid table to make information easier to process.

As a developer, you can use SAPUI5 adaptation projects in [SAP Business Application Studio](#) to create applications variants. You can efficiently reuse the original SAP Fiori application together with its OData service and application logic to define changes that are specific to the application variant only.

After deployment, both the base application and the newly created application variant exist and can be made available to the relevant business users through corresponding tiles in the SAP Fiori launchpad. If the base application is changed by a new version, it does not affect the deployed application variant. You can test the application variant with the latest enhancements of the base application in SAP Business Application Studio and deploy it again to benefit from the latest enhancements. For further information, see [Check Whether Your Adaptation Project Is Up-To-Date with Base App Upgrades](#).

The following diagram shows the relationship between the source application (shown in blue) and the application variant created by an adaptation project (shown in dark blue).



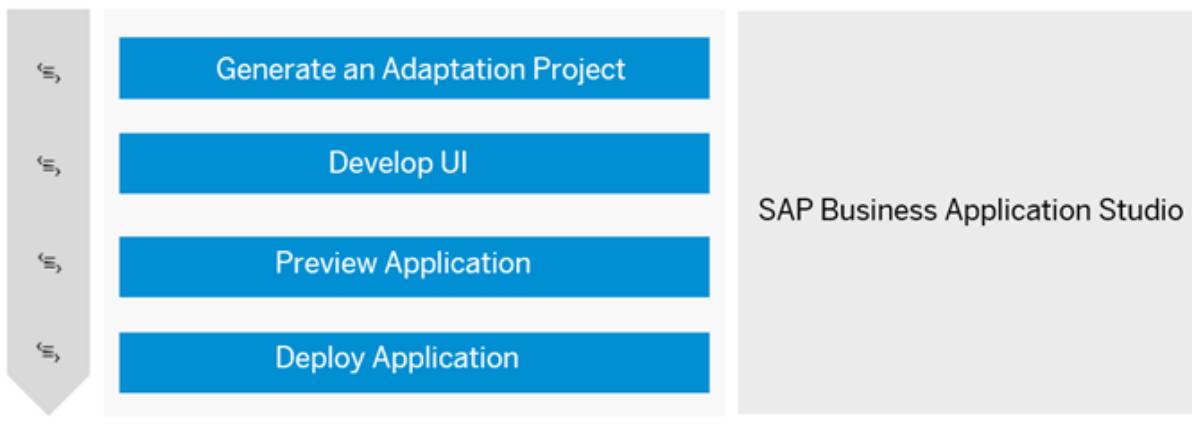
#### Prerequisites

You have set up and integrated SAP Business Application Studio. See [Setup of UI Development in SAP Business Application Studio](#).

## Generating and Deploying Your Adaptation Project

#### Prerequisites

- The following business catalogs are assigned to your user:
  - Business catalog `SAP_A4C_BC_DEV_OBJ_DIS_PC` (or other business catalog for development tasks), which allows you to retrieve the list of extensible base applications.
  - A business catalog that grants access to the base application, which allows you to execute the corresponding OData service.
  - Business catalog `SAP_A4C_BC_DEV_UID_PC`, which allows you to deploy the adaptation project.
- You have an open transport request.

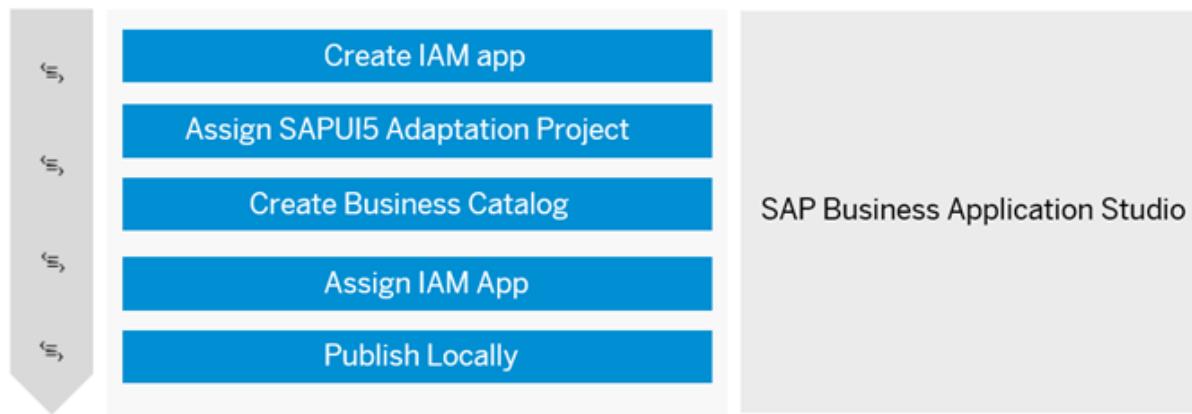


#### Legend

↳ Task for Developers

1. As a developer user in SAP Business Application Studio, generate an SAPUI5 adaptation project. See [Create the Adaptation Project](#).
2. Make adaptations to the UI, for example, change control properties, add own sections to the object page, add controller extensions, etc.. See [Make Adaptations](#).
3. Now you can preview the generated SAP Fiori application. See [Preview the Adaptation Project](#).
4. Deploy the SAP Fiori UI by executing command `npm run deploy` in the terminal of your project. See [Deploy or Update the Adaptation Project to the ABAP Repository](#) .

## Creating and Publishing Your Identity and Access Management (IAM) App



#### Legend

↳ Task for Developers

1. To manage access to your SAP Fiori application, you need to log on to the development tenant in ABAP Development Tools for Eclipse as a developer to create an Identity and Access Management (IAM) application. Moreover, you need to assign an SAPUI5 adaptation project and maintain authorizations, as shown in steps one and two in the figure above. See [Creating an IAM App for the Business Service \(Developer\)](#).

1. Open and log onto ABAP Development Tools (ADT).
2. In the Project Explorer, open your package and expand the BSP Library folder and then the BSP Applications folder to view the application variant you deployed in SAP Business Application Studio.
3. In the same package, you will find the folder Fiori User Interface containing the FLP App Descriptor Items folder. An FLP app descriptor item with the name <BSP application name>\_UI5R was generated automatically when you deployed the application variant in SAP Business Application Studio.
4. Right-click on your package and choose New -> Other ABAP Repository Object
5. In the ABAP Repository Object dialog box, open the Cloud Identity and Access Management folder and select IAM App.
6. Choose [Next](#).
7. In the New IAM App dialog box, enter a unique name and description for your IAM app.
8. In the Application Type field, select UI5A – UI Adaptation App. The application ID suffix \_UI5A is set automatically.
9. Choose [Next](#)
10. In the Select Transport Request dialog box, choose the transport request you want to use and choose [Finish](#).
11. In your package, expand the folder Cloud Identity and Access Management -> IAM Apps and you will see your newly created IAM app with the application name you specified in step g) and the suffix UI5A.
12. Open your newly created IAM App in the IAM Apps folder. In the [Overview](#) tab, enter the *Fiori Launchpad App Description Item ID* that was generated automatically in your *Fiori User Interface folder -> FLP Descriptor Items* called <BSP application name>\_UI5R.
13. Open your newly created IAM App in the IAM Apps folder. On the [Overview](#) tab, enter the *Fiori Launchpad App Description Item ID* that was generated automatically in your *Fiori User Interface folder -> FLP Descriptor Items* called <BSP application name>\_UI5R.
14. Open the [Services](#) tab, which will now list the services used by your application variant. Choose [Synchronize](#).
15. Open the [Authorizations](#) tab and maintain authorizations for your application variant and choose [Save](#).
16. You need to publish your IAM app locally, to do this choose the [Publish Locally](#) button on the top right of the screen.
2. Once you have created your IAM app, you have to create a business catalog. See [Creating a Business Catalog \(Developer\)](#).
3. Assign your IAM app to the catalog.
4. Publish the IAM app and business catalog locally.

## Next Steps

[Add Your App to SAP Fiori Launchpad](#)

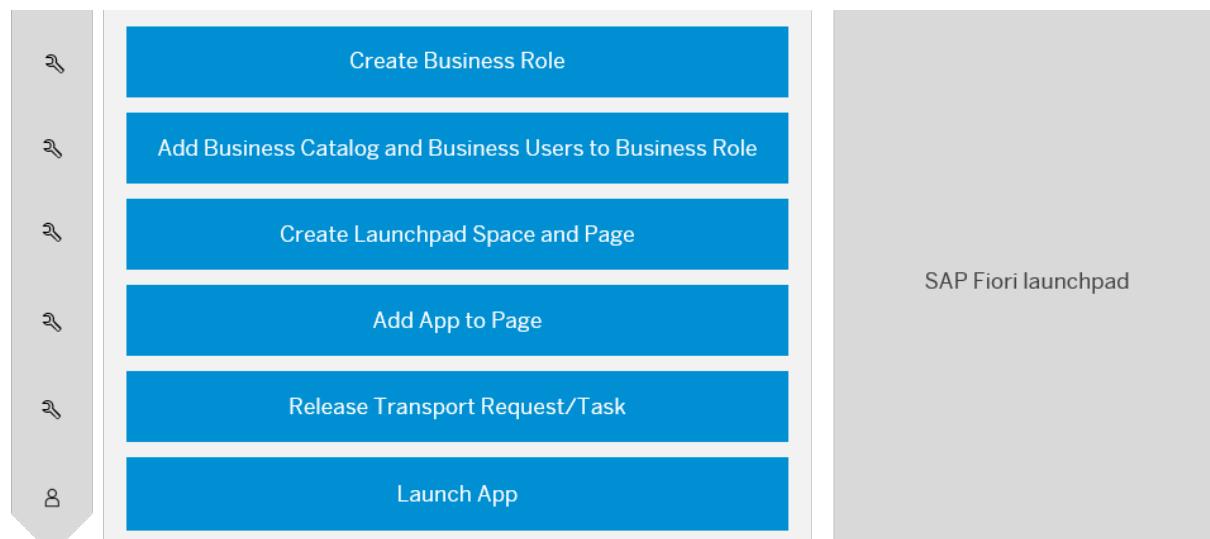
## Related Information

[Extending an SAP Fiori Application for S/4HANA Cloud public Edition and SAP BTP, ABAP Environment](#)  
[SAP Business Application Studio](#)

#### 4.2.12.1.4 Add Your App to SAP Fiori Launchpad

Get an overview on how to add your app to SAP Fiori launchpad.

If you need further assistance with the individual steps, check out the developer tutorial [Integrate List Report into ABAP Fiori Launchpad](#).



##### Legend

- Task for Administrators
- Task for Business Users

- In the SAP Fiori launchpad of your ABAP system, create a business role with the [Maintain Business Roles](#) app. See [Maintain Business Roles](#).
- Assign a business catalog and business user to the business role. See [Maintain Business Users](#).
- Create a launchpad space and page. See [Step by Step: Create a New Space and Page for a Business Role](#).

##### ⓘ Note

To use this app, your user must be assigned to business catalog SAP\_CORE\_BC\_UI\_FLD.

- Enable spaces for your user. See [Enabling Spaces](#). Optionally, you can personalize and adapt the user interface of your app for all users. See [Personalizing and Adapting Apps](#).
- As a business user, you can launch the app.

### Next Step

[Transport Your App](#)

## 4.2.12.1.5 Scoping Space and Page Templates

The scoping of space and page templates refers to the process of defining the visibility or accessibility of these objects.

Space and page templates are not visible in the [Manage Launchpad Spaces](#) and [Manage Launchpad Pages](#) apps in the development system by default. Meaning, they cannot be assigned or used within a business role. The space and page templates need to be scoped.

To use the space and page templates in the development system you must implement your own scoping using the scoping API (Application Programming Interface) `DL APS BC SCOPE CHANGE API`. It's recommended to use a console application to set the space and page template in scope.

### ↔ Sample Code

```
CONSTANTS obj_name_page  TYPE c LENGTH 40 VALUE 'PAGE_NAME' ##NO_TEXT.
CONSTANTS obj_name_space TYPE c LENGTH 40 VALUE 'SPACE_NAME' ##NO_TEXT.
DATA(lo_scope_api) = cl_aps_bc_scope_change_api->create_instance( ).
lo_scope_api->scope(
    EXPORTING it_object_scope = VALUE #(
        pgmid      = if_aps_bc_scope_change_api->gc_tadir_pgid-R3TR
        scope_state = if_aps_bc_scope_change_api->gc_scope_state-ON
        ( object = if_aps_bc_scope_change_api->gc_tadir_object-UIST
    obj_name = obj_name_space )
        ( object = if_aps_bc_scope_change_api->gc_tadir_object-UIPG
    obj_name = obj_name_page ) )
        iv_simulate      = abap_false
        iv_force         = abap_false
    IMPORTING et_object_result = DATA(lt_results)
        et_message       = DATA(lt_messages) .
LOOP AT lt_results INTO DATA(ls_result).
    ...
ENDLOOP.
```

In non-development systems, space and page templates are scoped automatically during import into the system.

## Related Information

[Scoping Framework](#)

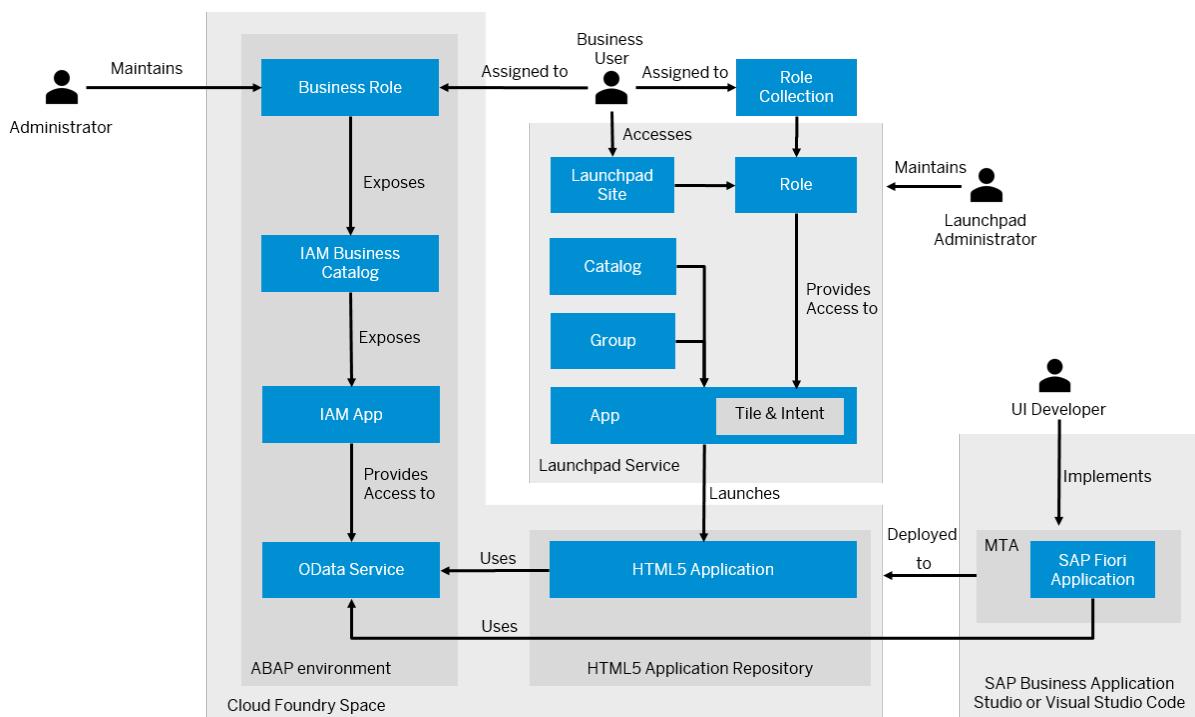
## 4.2.12.2 SAP Fiori Applications in the Cloud Foundry Environment

When you plan to create an SAP Fiori application that you have deployed to the SAP BTP, Cloud Foundry environment and want to provide this application to business users, you may want to get a better understanding of the required steps before getting started with the UI development.

### → Recommendation

We recommend using the same UI5 version that is being used by the ABAP environment system.

To check the UI5 version in the SAP Fiori launchpad of your ABAP environment system, use command **Alt+Ctrl+Shift+P** or navigate to the user actions menu and select *About*. See [User Actions Menu](#).



## Implementing and Deploying the UI

To develop an SAP Fiori application, you can create a multitarget application using the Fiori Tools application router generator, where we recommended using the SAP-managed approuter. See [Application Router](#) and [Developing HTML5 Applications and Extensions](#).

Afterwards, you can start creating your SAP Fiori application using the Fiori Tools application generator and add it to the multitarget application (MTA). When generating the application, you can browse the available OData services for UI development. The application generation process also includes adding information for the SAP Fiori app tile in the SAP Fiori launchpad, such as a title, action, and semantic object. This information has to be defined in the project attributes of your MTA by adding an FLP configuration. Your configuration is then stored in the `manifest.json` file. Note that only one intent navigation consisting of a semantic object and action is supported. If you change the ID of the intent, existing references from the SAP Fiori launchpad to the tile break.

You can integrate the OData service of your ABAP environment into the application by specifying a corresponding route in the `xs-app.json` file. The routing can be done either by providing a destination that points to the ABAP environment instance or by specifying the service endpoint of the ABAP environment and binding the service instance. See [Configure Destinations](#) and [Binding Service Instances to Applications](#). The Fiori application generator adds a route to the `xs-app.json` file using the destination that you specify in the generator.

### Note

If you have used the Fiori Tools application generator to create your app, you have selected a destination that points to your development subaccount. Make sure that the destination in your target subaccount has the same name as the one maintained in the `xs-app.json` file.

The next step is to build the MTA project. This generates an `*.mtar` file and `mta_archives` folder.

Before deploying your UI, you should preview your application. To preview your application or discover available OData services, business catalog `SAP_CORE_BC_EXT_TST` has to be assigned to your user, which allows you to preview your application and . See [Business Catalogs for Development Tasks \[page 2548\]](#).

The final step is to create a deployment configuration and deploy the MTA to your SAP BTP, Cloud Foundry environment space, which requires the [space developer role](#). By doing so, the SAP Fiori application that you have added to the MTA gets deployed to the HTML5 application repository.

## Providing Access to Business Users

After you have deployed the SAP Fiori application, you have to create an IAM app to make your application consumable for business users. This requires assigning the business service to the IAM app. Additionally, you have to use SAP Build Work Zone, standard edition and log on as an administrator to create a role.

A corresponding role collection is automatically created in the SAP BTP cockpit. This allows you to assign business users to the role collection in the SAP BTP cockpit. Being assigned to the role collection, business users can access the launchpad site and the app.

## Integrating the SAP Fiori App into SAP Fiori launchpad

To make an SAP Fiori application accessible on the SAP Fiori launchpad home page of business users, you have to assign your app to a group and catalog. See [Assign Apps to a Group and to a Catalog](#). Afterwards, you need to add content to the HTML5 apps content provider in SAP Build Work Zone, standard edition. The HTML5 apps provider reflects all the HTML5 apps that are deployed to the HTML5 app repository that is assigned to the subaccount. See [HTML5 Apps Content Provider](#) and [Integrate Apps and Shell Plugins](#).

## Related Information

[Develop an SAP Fiori Application and Deploy it to Cloud Foundry Using SAP Business Application Studio \[page 1855\]](#)

[Develop an SAP Fiori Application and Deploy it to Cloud Foundry Using Visual Studio Code \[page 1857\]](#)

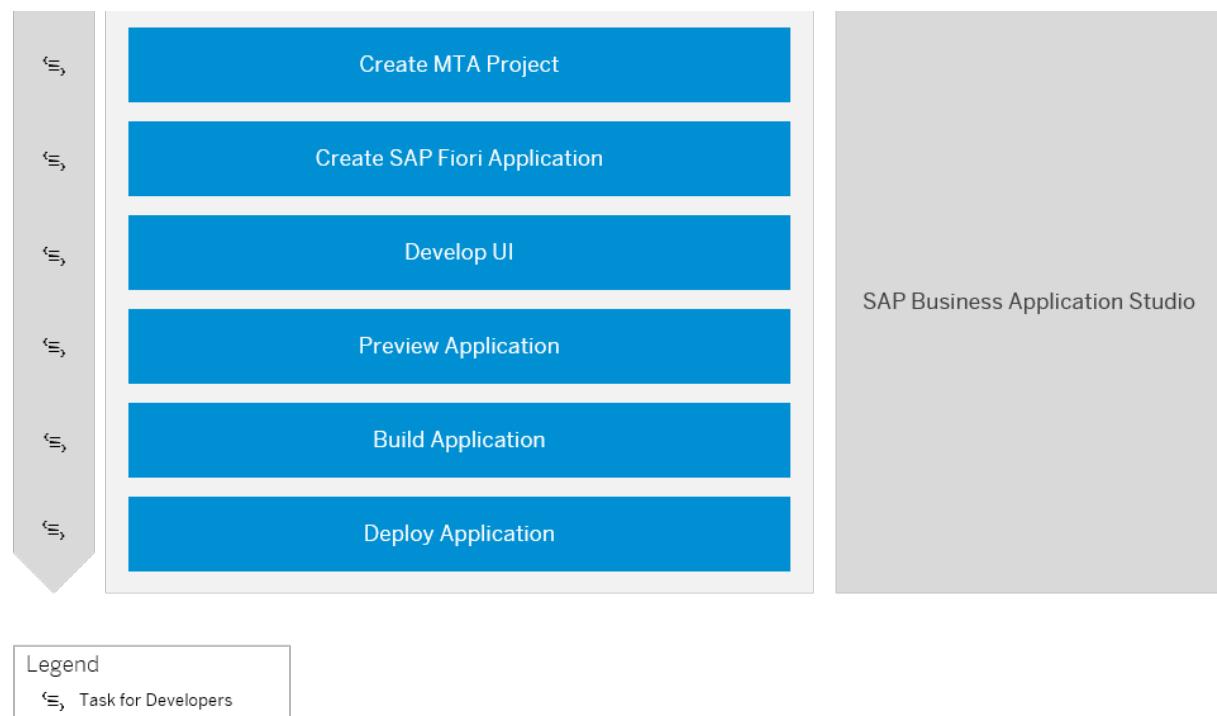
## 4.2.12.2.1 Develop an SAP Fiori Application and Deploy it to Cloud Foundry Using SAP Business Application Studio

Get an overview about how to create and deploy an SAP Fiori application to Cloud Foundry using SAP Business Application Studio.

### Prerequisites

- A subaccount administrator has provided access to SAP Business Application Studio. See [Setup of UI Development in SAP Business Application Studio \(Optional\) \[page 194\]](#).
- A subaccount administrator has created a destination to the ABAP system to integrate SAP Business Application Studio. See [Creating a Destination to the ABAP System for SAP Business Application Studio \[page 198\]](#).
- You have created a dev space of type SAP Fiori. See [SAP Business Application Studio](#).
- You have access to a RAP business service that has been exposed as an OData service. See [Using Service Binding Editor for OData V2 Service](#).

### Generating and Deploying Your Application



1. Create an MTA project with approuter configuration.

To do so, use the *Fiori CF Application Router Generator*. See [Generate an MTA Deployment File](#).

This generates an MTA project and the corresponding application router configuration.

2. With the *SAP Fiori Generator*, create an SAP Fiori application in the subfolder of the MTA file location by selecting *Fiori: Open Application Generator* from the command palette. See [Generate an Application](#) and [Add a Fiori Application to an MTA Deployment File with the Fiori Generator](#).
  - As a data source, select *Connect to a System*.
  - As a system, choose the destination to the ABAP system that has been created to integrate SAP Business Application Studio.
  - As a service, select your RAP business service that has been exposed as an OData serviceIn the *Project Attributes* section, add an SAP Fiori launchpad configuration for your UI project. See section *Add FLP configuration* in [Additional Configuration](#).  
If you want to create your FLP configuration later, see [SAP Fiori Launchpad Configuration](#).
3. Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).
4. Now you can preview the generated SAP Fiori application. See [Preview an Application](#).
5. Build the application by executing command `npm run build` in the terminal of your project. See [Deployment to Cloud Foundry](#).  
This generates an `*.mtar` file and `mta_archives` folder.
6. Log on to Cloud Foundry and deploy the SAP Fiori UI to your development space by executing command `npm run deploy`. See [Deployment to Cloud Foundry](#).

→ Tip

If you can't discover an available OData service in the generator, or preview your application, check if your user is assigned to business catalog `SAP_CORE_BC_EXT_TST`. If you aren't authorized to deploy the application, check if business catalog `SAP_A4C_BC_DEV_UID_PC` is assigned to your user.

## Related Information

[SAP Business Application Studio](#)

[SAP Fiori Tools](#)

[SAP Fiori Overview](#)

[Tutorial: Create an SAP Fiori App and Deploy it to SAP BTP, Cloud Foundry environment](#) 

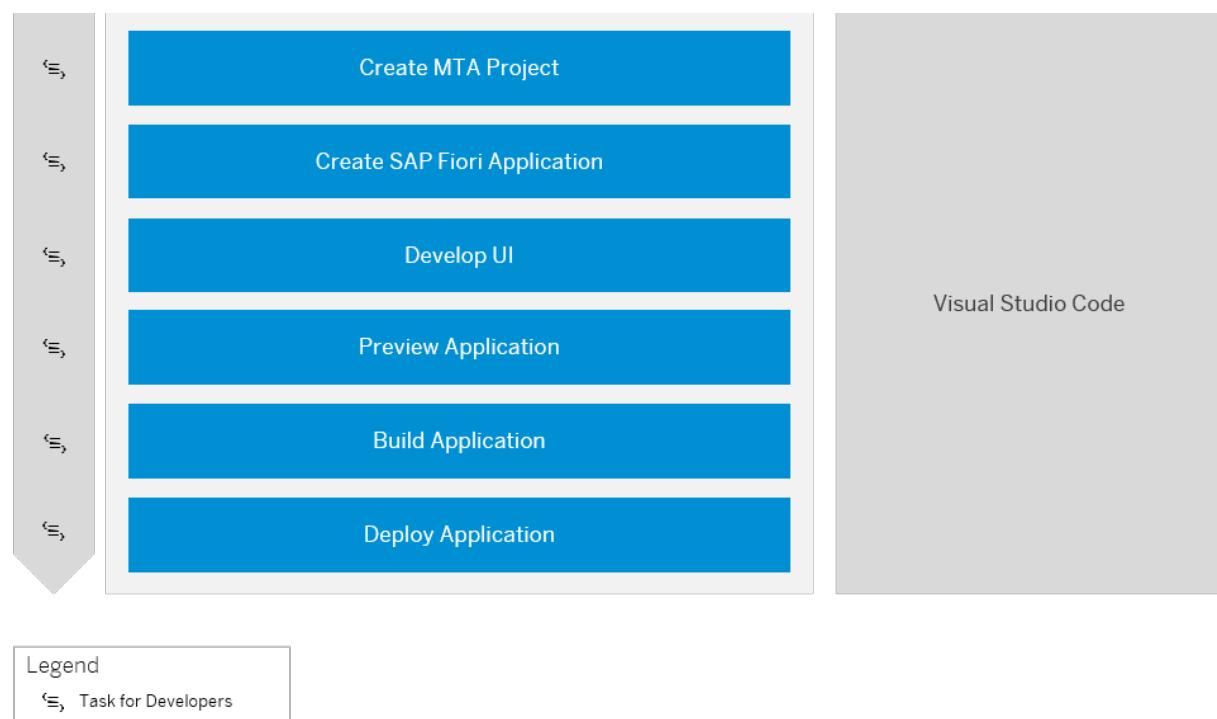
## 4.2.12.2.2 Develop an SAP Fiori Application and Deploy it to Cloud Foundry Using Visual Studio Code

Get an overview about how to create and deploy an SAP Fiori application to Cloud Foundry using Visual Studio Code (VS Code).

### Prerequisites

- You have installed and set up Visual Studio Code including the SAP Fiori tools extensions. See [Visual Studio Code](#).
- You have access to a RAP business service that has been exposed as an OData service. See [Using Service Binding Editor for OData V2 Service](#).
- To establish a connection with your ABAP environment system, you either have to be a space developer in the ABAP environment instance or have access to a service key in the ABAP environment instance. See [Add Space Members Using the Cockpit](#) and [Creating Service Keys](#) in the ABAP service instance.

### Generating and Deploying Your Application



1. Create an MTA project with approuter configuration.  
To do so, use the [Fiori CF Application Router Generator](#). See [Generate an MTA Deployment File](#).  
This generates an MTA project and the corresponding application router configuration.

- With the *Fiori Application Generator*, create an SAP Fiori application in the subfolder of the MTA file location by selecting *Fiori: Open Application Generator* from the command palette. See [Generate an Application](#) and [Add a Fiori Application to an MTA Deployment File with the Fiori Generator](#).

In the *Data Source and Service Selection* section of the Template Wizard, select the following values:

- Data source: *Connect to a System*.
- System: *New System*
- System type: *ABAP Environment on SAP Business Technology Platform*
- ABAP environment definition source:
  - (Option 1) **Discover a Cloud Foundry Service**

 **Note**

You have to log on to your Cloud Foundry space by executing command `cf login`. When you're prompted to enter the API endpoint and org name, you can navigate to your subaccount in the SAP BTP cockpit, where you can find this information.

- (Option 2) **Upload a Service Key File**
  - Continue with choosing a system name and service.
- In the *Project Attributes section*, add an SAP Fiori launchpad configuration for your UI project. See [Add FLP Configuration](#).
- Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).
  - Now you can preview the generated SAP Fiori application. See [Preview an Application](#).
  - Build the application by executing command `npm run build` in the terminal of your project. See [Deployment to Cloud Foundry](#).  
This generates an `*.mtar` file and `mta_archives` folder.
  - `npm run deploy`. See [Deployment to Cloud Foundry](#).

 **Tip**

If you can't discover an available OData service in the generator, or preview your application, check if your user is assigned to business catalog Log on to Cloud Foundry and deploy the SAP Fiori UI to your development space by executing command `SAP_CORE_BC_EXT_TST`. If you aren't authorized to deploy the application, check if business catalog `SAP_A4C_BC_DEV_UID_PC` is assigned to your user.

## Related Information

[SAP Fiori Tools](#)

[SAP Fiori Overview](#)

### 4.2.12.2.3 Integration with Approuter

You can integrate the OData service of your ABAP environment into the application by specifying a corresponding route in the `xs-app.json` file of the HTML5 module. The routing can be done either by

providing a destination that points to the ABAP environment instance or by specifying the service endpoint of the ABAP environment and binding the service instance.

#### Note

If the application and the instance of the ABAP environment are deployed to different subaccounts, use routing via destination.

For more information about the two types of routing, navigate to [Routing via Service \[page 1859\]](#) or [Routing via Destination \[page 1861\]](#).

### 4.2.12.2.3.1 Routing via Service

#### Maintain Route

In order to access the service instance of the ABAP environment directly, maintain a route with service com.sap.cloud.abap and endpoint abap for the path of the OData service in the xs-app.json file.

#### Sample Code

```
"routes": [
  {
    "source": "^/sap/(.*)$",
    "target": "/sap/$1",
    "service": "com.sap.cloud.abap",
    "endpoint": "abap",
    "authenticationType": "xsuaa",
    "csrfProtection": false
  }
]
```

#### Bind ABAP Service Instance

Bind the service instance to the approuter. You can bind the service instance using the SAP BTP cockpit or by supplying the information in the multitarget application.

The following service binding parameters are supported by the ABAP service:

Binding Parameter	Description
abap_endpoint_timeout	Positive integer that defines the time in milliseconds when the communication between the approuter and service instance will be timed out. The default value is 30000, which corresponds to 30 seconds.  This default value is suitable for most applications. However, the communication with the service instance may take longer. If you experience timeouts, you can maintain a value higher than 30 seconds. The maximum value is 600000, which corresponds to 10 minutes.
create_x_sap_security_session	Boolean that defines if the approuter shall request a security session cookie by setting the <code>x-sap-security-session: true</code> http header. The default is true.  For more information regarding security session cookies, see <a href="#">SAP Note 3319136 - No implicit security session creation for API callers calling an ABAP Cloud system as of 2402</a> .

For more information, see [Binding Service Instances to Applications](#).

## Bind Service Instance Using the Cockpit

- Log on to the cockpit and go to the subaccount that contains the space you'd like to navigate to. See [Navigate in the Cockpit](#).
- In the navigation area, go to **Cloud Foundry Spaces** and choose your space.
- Select **service instances** from the navigation area and navigate to your ABAP system service instance.
- In the service instance details section that opens to the right, select the **Actions** menu and then select **Bind Application**.
- In the **New Binding wizard**, choose the application to bind.
- Provide the parameters for your binding by specifying the parameter in JSON format. For example:

### ↔ Sample Code

```
{  
  "abap_endpoint_timeout": 600000,  
  "create_x_sap_security_session": false  
}
```

For more information, see [Binding Service Instances to Applications](#).

## Bind Service Instance in Multitarget Application

To bind the service instance to the approuter, add the service instance as a required resource in the `mta.yaml` file. For example:

### Sample Code

```
_schema-version: "3.2"
ID: my-mta
version: 1.0.
modules:
- name: my-app
  type: approuter.nodejs
  path: cf/router
  requires:
  - name: my-abap-service-instance
    parameters:
      config:
        create_x_sap_security_session: false
        abap_endpoint_timeout: 600000
    ...
resources:
- name: my-abap-service-instance
  type: org.cloudfoundry(existing-service
parameters:
  service: abap
  service-plan: standard
  service-name: my-abap-service-instance-name
...
```

### 4.2.12.2.3.2 Routing via Destination

## Maintain Route

To access the service instance of the ABAP environment via a destination, maintain a route for the path of the OData service with a destination that points to the ABAP environment instance in the `xs-app.json` file.

### Sample Code

```
"routes": [
  {
    "source": "^/sap/(.*)$",
    "target": "/sap/$1",
    "destination": "my-destination",
    "authenticationType": "xsuaa",
    "csrfProtection": false
  }
]
```

## Bind Destination Service Instance

Bind the service instance to the approuter. You can bind the service instance using the cockpit or by supplying the information in the MTA. See [Binding Service Instances to Applications](#) for more information.

To bind the destination service instance to the approuter in the multitarget application, add it as a required resource in the `mta.yaml` file. For example:

### Sample Code

```
_schema-version: "3.2"
ID: my-mta
version: 1.0.
modules:
- name: my-app
  type: approuter.nodejs
  path: cf/router
  requires:
  - name: my-abap-service-instance
    parameters:
      config:
        create_x_sap_security_session: false
        abap_endpoint_timeout: 600000
...
resources:
- name: my-abap-service-instance
  type: org.cloudfoundry(existing-service
parameters:
  service: abap
  service-plan: standard
  service-name: my-abap-service-instance-name
...
...
```

## Maintain Destination

Create a destination with the same name as specified in the `xs-app.json` file in the SAP Destinations Service instance that is bound to your application or on subaccount level, for example, maintain the `my-destination` destination in the `my-destination-service-instance-name` SAP Destination service instance.

If the application and the instance of the ABAP environment reside in the same subaccount, use the `OAuth2UserTokenExchange` authentication type for the destination for principal propagation. If the application and the instance of the ABAP environment are deployed to different subaccounts, then use the `SAMLAssertion` authentication type.

You can add additional properties to the destination, for instance:

Property	Description
HTML5.Timeout	Positive integer representing the maximum time to wait for a response (in milliseconds) from the destination. The default value is 30000 milliseconds, which corresponds to 30 seconds.

Property	Description
URL.headers.<header-key>	<p>A static key prefix used as a namespace grouping of the URL's HTTP headers whose values will be sent to the target endpoint.</p> <p>For instance, if you maintain the property <code>URL.headers.x-sap-security-session</code> with value <code>true</code>, then the appropriate will send the <code>x-sap-security-session: true</code> http header.</p> <p>For more information regarding security session cookie, see <a href="#">3319136</a>.</p>

## Related Information

[Application Routes and Destinations](#)

## 4.2.13 ABAP Service Instance URL

### Context

Previously, when creating a new ABAP Cloud project in ABAP development tools for Eclipse, a service key was necessary to connect to the SAP BTP, ABAP environment. In the window [Connection to an ABAP Service Instance](#), there was a distinction between the connection to SAP S/4HANA Cloud Public Edition and SAP BTP, ABAP environment. From now on, only one option to paste in your ABAP service instance URL is available.

### Connection to an ABAP Service Instance

1. Select [Create a new ABAP cloud project](#). This opens up a window called [Connection to an ABAP Service Instance](#).
2. Following, you can now choose two options to connect to your ABAP service instance listed below.
  1. Insert the URL from the SAP Fiori launchpad of your system.
  2. If you still have a service key available, you can extract the URL from it by clicking the link shown in the description text of the window you're in: **Do you have a service key? Extract the URL from the service key.**
    1. In this case, a new window opens up where you can paste in your service key.
    2. Next, choose [Copy to Clipboard](#).
    3. Now, you are in the [Connect to an ABAP Service Instance](#) window again and can paste in the service instance URL.
    4. Choose [Copy logon URL to clipboard](#).

- Finally, you can paste the URL into your browser and log on to the system.
- The message shows *You have been successfully logged on* and ABAP development tools for Eclipse will pick up your project data.

## 4.3 Development in the Kyma Environment

Learn more about developing applications in SAP BTP, Kyma runtime.

### Overview

With SAP BTP, Kyma runtime, you can build simple Functions, develop, and deploy more complex microservices, or mixtures of those, depending on your use case complexity level. Functions and microservices can act as stand-alone applications or extensions of these SAP systems:

- SAP S/4HANA
- SAP S/4HANA Cloud
- SAP S/4HANA Cloud, public edition
- SAP Field Service Management
- SAP SuccessFactors
- SAP Customer Experience systems:
  - SAP Commerce Cloud
  - SAP Cloud for Customer
  - SAP Marketing Cloud

### Language of Choice

With the Kyma runtime, you can create microservices in the language of your choice and run them as containerized applications. Additionally, you can create Functions in Node.js or Python.

### Kyma Dashboard and kubectl

The Kyma runtime comes with a central administration dashboard (Kyma dashboard), which allows you to deploy microservices, create Functions, and manage their configurations. You can also use it to connect SAP BTP services to your cluster and manage them using SAP BTP Service Operator, create instances of these services, and use them in your microservices or Functions.

For those who prefer to work with command-line tools, the Kyma runtime also offers the Kubernetes command-line tool, kubectl.

## SAP Cloud Application Programming Model

The SAP Cloud Application Programming Model (CAP) is the recommended framework for application and service development in the Kyma runtime. To learn more, see [Programming Models \[page 124\]](#).

## Using API

The APIs of Kyma modules and third-party modules' may vary from the least stable alpha version through a more stable beta and the most stable vX version (X being an integer). While using the alpha version, you can face issues with module upgrades, because there is no guarantee that the API will not change or won't be removed in the future. For more information, check the [Kubernetes API versioning](#).

→ Tip

To get the list of the APIs, their kind, and version, run:

```
kubectl get crd -o custom-columns="KIND:.spec.names.kind, GROUP:.spec.group, VERSION:.spec.versions[*].name"
```

If an API is available in two versions, always use the version that is more stable.

## Development Tutorials

To help you get started with the development process, go through the following tutorials that show how to [develop a full-stack application](#) in the Kyma runtime. You'll create a sample containerized microservice, learn how to expose it via API, and trigger it with events. Additionally, you create Functions as well as instances of external services that you can use in your microservices and Functions.

- [Deploy MSSQL in the Kyma Environment](#)
- [Deploy a Go MSSQL API Endpoint in the Kyma Environment](#)
- [Deploy the SAPUI5 Frontend in the Kyma Environment](#)
- [Trigger a Microservice with an Event](#)
- [Use Redis in the Kyma Environment to Store and Retrieve Data](#)

## Related Information

- [Extension Samples](#)  
Kyma provides a ready-to-use project that contains sample applications. You can use them to build event- and API-based extensions in the Kyma runtime using your favorite technology. These samples are implemented in multiple languages and demonstrate various Kyma features and use case scenarios.
- [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#)  
With the Kyma runtime, you can extend the SAP systems to build and deploy your own applications.

- [Deploy Workloads in the Kyma Environment to Extend SAP Systems \[page 1977\]](#)  
Access the Kyma runtime and start creating extensions for SAP systems using Functions.

## 4.3.1 Kyma Modules

With Kyma's modular approach, you can install just the modules you need, instead of a predefined set of components.

You can choose to add any modules as required. To learn how, see [Add and Delete a Kyma Module \[page 3012\]](#). To find out which module version is running in your cluster, go to [Kyma dashboard](#).

### → Tip

A release of a new module's version is announced with a release note in [What's New for SAP Business Technology Platform](#) for both, the fast and regular channels:

- On the day of the release in the fast channel, a release note is published with the *Preview* label.
- When the module version becomes available in the regular channel (after approximately two weeks), the *Preview* label is removed.

## Default Kyma Modules

When you create Kyma runtime in SAP BTP cockpit, it is provisioned with the default modules added. The default modules are not mandatory. If you don't need them, you can delete them in [Kyma dashboard](#). See [Add and Delete a Kyma Module \[page 3012\]](#).

Default Kyma Modules

Module	Technical name	Purpose	Documentation
<a href="#">API Gateway</a>	<code>api-gateway</code>	API Gateway provides functionalities that allow you to expose and secure APIs.	<a href="#">kyma-project.io: API Gateway module</a>
<a href="#">Istio</a>	<code>istio</code>	Istio is a service mesh with Kyma-specific configuration.	<a href="#">Istio Module [page 1868]</a>
<a href="#">SAP BTP Operator</a>	<code>btp-operator</code>	Within the SAP BTP Operator module, BTP Manager installs the SAP BTP service operator that allows you to consume SAP BTP services from your Kubernetes cluster using Kubernetes-native tools.	<a href="#">SAP BTP Operator Module [page 1894]</a> <a href="#">What's New for SAP BTP Operator</a> <a href="#">GitHub repository: BTP Manager</a>

## Optional Kyma Modules

When you create Kyma runtime in SAP BTP cockpit, the following modules are not added by default, but you can choose to add and delete them anytime in [Kyma dashboard](#). See [Add and Delete a Kyma Module \[page 3012\]](#).

### Optional Kyma Modules

Module	Technical name	Purpose	Documentation
<i>Application Connector</i>	<code>application-connector</code>	Application Connector allows you to connect with external solutions. No matter if you want to integrate an on-premise or a cloud system, the integration process doesn't change, which allows you to avoid any configuration or network-related problems.	<a href="#">kyma-project.io: What is Application Connectivity in Kyma?</a> ↗ <a href="#">GitHub repository: Application Connector Manager</a> ↗
<i>Eventing</i>	<code>eventing</code>	<p>Eventing provides functionality to publish and subscribe to CloudEvents.</p> <p>At the moment, the SAP Event Mesh default plan and NATS are supported. If you choose NATS, add the NATS module.</p>	<a href="#">Configure SAP Event Mesh for Kyma Eventing [page 1986]</a> <a href="#">kyma-project.io: Eventing module</a> ↗ <a href="#">GitHub repository: Eventing</a> ↗
<i>Keda</i>	<code>keda</code>	The Keda module comes with Keda Manager, an extension to Kyma that allows you to install <a href="#">KEDA</a> ↗ (Kubernetes Event Driven Autoscaler).	<a href="#">Keda Module [page 1918]</a>
<i>NATS</i>	<code>nats</code>	NATS deploys a NATS cluster within the Kyma cluster. You can use it as a backend for Kyma Eventing.	<a href="#">kyma-project.io: NATS module</a> ↗ <a href="#">GitHub repository: NATS</a> ↗
<i>Serverless</i>	<code>serverless</code>	With the Serverless module, you can define simple code snippets (Functions) with minimal implementation effort.	<a href="#">Deploy Workloads in the Kyma Environment to Extend SAP Systems [page 1977]</a> <a href="#">kyma-project.io: What is Serverless in Kyma?</a> ↗ <a href="#">kyma-project.io: Serverless Configuration</a> ↗ <a href="#">GitHub repository: Serverless</a> ↗
<i>Telemetry</i>	<code>telemetry</code>	The Telemetry module collects application logs and distributed traces for your application, and dispatches them to your preferred backends.	<a href="#">Telemetry Module [page 1921]</a> <a href="#">What's New for Telemetry</a> <a href="#">GitHub repository: Telemetry</a> ↗

## Other SAP Modules

These modules are developed and maintained by SAP teams outside of Kyma. To get help or request a feature, contact the module provider directly.

Module	Technical name	Purpose	Documentation
<i>Transparent Proxy</i>	<code>transparent-proxy</code>	Use the transparent proxy for Kubernetes to connect workloads in a Kubernetes cluster to Internet and on-premise applications.	<a href="#">Transparent Proxy in the Kyma Environment</a>
<i>Connectivity Proxy</i>	<code>connectivity-proxy</code>	Use the connectivity proxy for Kubernetes to connect workloads in a Kubernetes cluster to on-premise systems, exposed via the Cloud Connector.	<a href="#">On-Premise Connectivity in the Kyma Environment</a>

### 4.3.1.1 Istio Module

Use the Istio module to manage and configure the Istio service mesh.

#### What Is Istio?

Istio is an open-source service mesh that provides a uniform way to manage, connect, and secure microservices. It helps to manage traffic, enhance security capabilities, and provide telemetry data for understanding service behavior. See the [open-source Istio documentation](#).

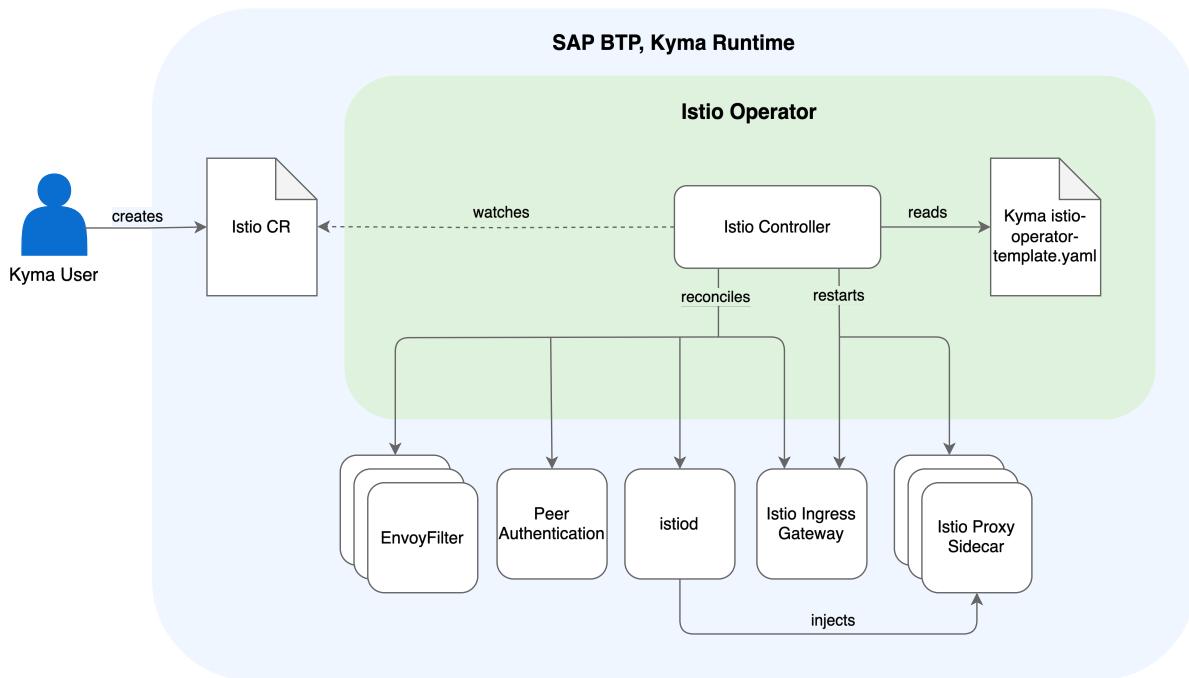
The Istio module installs and manages Istio in your Kyma cluster. By default, the Istio module is automatically added once you create a Kyma runtime instance.

#### Features

The Istio module offers the following features:

- Management of Istio installation and upgrades: The module installs Istio and simplifies the process of managing its installation, reducing the complexity and time required for maintenance.
- Default Istio configuration: You can quickly have Istio installed with default settings.
- Fine-tuning capabilities: You can optimize Istio and fine-tune its settings according to specific performance or operational requirements.
- Synchronization of the data plane with the Istio control plane: This ensures that changes you make to the control plane are consistently reflected in the data plane, ensuring that the network operates consistently and reliably without any discrepancies.
- Support for the X-Forwarded-For (XFF) header: You can configure the XFF header to manage and track the source of incoming requests.

## Architecture



### Istio Operator

Within the Istio module, Istio Operator handles the management and configuration of the Istio service mesh. It contains one controller, referred to as Istio Controller.

### Istio Controller

Istio Controller manages Istio installation as defined in the Istio custom resource (CR). Istio Controller is responsible for:

- Installing, upgrading, and uninstalling Istio
- Restarting workloads that have Istio sidecar proxy injected to ensure that these workloads use the correct version of Istio

## API/Custom Resource Definitions

The `istios.operator.kyma-project.io` CustomResourceDefinition (CRD) describes the Istio CR that Istio Controller uses to manage the installation of Istio. See [Istio Custom Resource](#).

## Resource Consumption

To learn more about the resources used by the Istio module, see [Kyma Modules' Sizing \[page 3016\]](#).

### 4.3.1.1.1 Istio Sidecar Proxies

Learn more about Istio sidecar proxies: what they are, what benefits they bring, how and when the Istio module restarts workloads with Istio sidecar proxy injection enabled.

#### What Is a Service Mesh?

A service mesh is an infrastructure layer that handles service-to-service communication, proxying, service discovery, traceability, and security, independently of the code of the services. To deliver this functionality, the Istio module uses the [Istio service mesh](#) that is customized for the specific needs of an implementation. The main principle of the Istio service mesh is to inject Pods of every service with Istio sidecar proxy, which is an extended version of the Envoy proxy. Envoy intercepts the communication between the services and regulates it by applying and enforcing the rules you create.

#### Purpose and Benefits of Istio Sidecar Proxies

By default, Istio installed as part of the Istio module is configured with automatic Istio sidecar proxy injection disabled. This means that none of your workloads' Pods, except those in the `kyma-system` namespace, get their own sidecar proxy container running next to the application. When Istio sidecar proxy injection is disabled for a service or for a namespace, you must manage mutual TLS (mTLS) traffic in services or at a namespace level by creating [DestinationRule](#) and [PeerAuthentication](#) resources.

With an Istio sidecar proxy injected, a resource becomes part of the Istio service mesh, which brings the following benefits that would be complex to manage otherwise.

#### Secure Communication

The Istio module sets [peer authentication](#) to cluster-wide STRICT mode. This ensures that your workload only accepts [mutual TLS \(mTLS\) traffic](#) where both client and server certificates are validated to ensure that all traffic is encrypted. This provides each service with a strong identity and a reliable system for managing keys and certificates.

Also, with Istio sidecar proxy injected, you can perform [request authentication](#) for your service. Istio enables request authentication with JSON Web Token (JWT) validation using a custom authentication provider.

#### Observability

Istio sidecar proxies enhance tracing capabilities by performing global tracing and forwarding the data to a tracing backend using the [OTLP protocol](#). When you integrate your application into the Istio service mesh, you can easily access advanced observability features without needing to implement complex instrumentation within the application. See [Kyma Modules With Tracing Capabilities \[page 1945\]](#).

#### Traffic Management

If you have the sidecar injected into every workload, you can use Istio's traffic routing rules without additional configuration. See [Traffic management](#).

[Traffic shifting](#) and [request routing](#) allows you to use techniques like canary releases and A/B testing to make your software release process faster and more reliable. To improve the resiliency of your applications, you can use [mirroring](#) and [fault injection](#) for testing and audit purposes.

## Resiliency

Application resiliency is an important topic within traffic management. Traditionally, application libraries implemented resiliency features like timeouts, retries, and circuit breakers. However, you can delegate such tasks to a service mesh, and the same configuration options work regardless of the programming language of your application. See [Network Resilience and Testing](#).

## Restart of Workloads with Enabled Istio Sidecar Injection

When the Istio version is updated or the configuration of Istio sidecar proxies changes, the Pods that have Istio sidecar proxy injection enabled are automatically restarted. This is possible for all resources that allow for a rolling restart. If Istio is uninstalled, the workloads are restarted again to remove the Istio sidecar proxies. However, if a resource is a Job, a ReplicaSet that is not managed by any Deployment, or a Pod that is not managed by any other resource, the restart cannot be performed automatically. In such cases, a warning is logged, and you must manually restart the resources. The Istio module does not restart an Istio sidecar proxy if it has a custom image set. See [Resource Annotations](#).

### 4.3.1.1.1 Enabling Istio Sidecar Proxy Injection

To include your workloads into the Istio service mesh, enable the Istio sidecar proxy injection. You can do this for either an entire namespace or a single Deployment. Learn how to perform both these operations.

## Enabling Injection for a Namespace

### Prerequisites

- You have the Istio module added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- To use CLI instructions, you must install [kubectl](#). Alternatively, you can use Kyma dashboard.

### Context

Enabling Istio sidecar proxy injection for a namespace allows istiod to watch all Pod creation operations in this namespace and automatically inject newly created Pods with an Istio sidecar proxy.

### ⓘ Note

A Pod is not injected with an Istio sidecar proxy if:

- Istio sidecar proxy injection is disabled at the namespace level
- The `sidecar.istio.io/inject` label on the Pod is set to false
- The Pod's spec contains `hostNetwork: true`

## Procedure

- You can either use Kyma dashboard, or kubectl.
- Use Kyma dashboard.
  - a. Select the namespace where you want to enable Istio sidecar proxy injection.
  - b. Choose *Edit*.
  - c. In the *UI Form* section, switch the toggle to enable Istio sidecar proxy injection.
  - d. Choose *Save*.
- Use kubectl.

Replace the placeholder and run the command:

```
kubectl label namespace <namespace-name> istio-injection=enabled
```

## Results

You've enabled Istio sidecar proxy injection for the specified namespace. The namespace is labeled with `istio-injection: enabled`, which means that all Pods created in it from now on will have the Istio sidecar proxy injected.

## Enabling Injection for a Deployment

### Prerequisites

- You have the Istio module added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- To use CLI instructions, you must install `kubectl`. Alternatively, you can use Kyma dashboard.

### Context

Enabling Istio sidecar proxy injection for a Deployment injects an Istio sidecar proxy into all the Deployment's Pods.

### ⓘ Note

A Pod is not injected with an Istio sidecar proxy if:

- Istio sidecar proxy injection is disabled at the namespace level
- The `sidecar.istio.io/inject` label on the Pod is set to false
- The Pod's spec contains `hostNetwork: true`

## Procedure

- Use Kyma dashboard.
  - a. Select the namespace of the Deployment for which you want to enable Istio sidecar proxy injection.
  - b. In the [Workloads](#) section, choose [Deployments](#).
  - c. Choose the Deployment.
  - d. Choose [Edit](#).
  - e. In the [UI Form](#) section, switch the toggle to enable Istio sidecar proxy injection.
  - f. Choose [Save](#).
- Use kubectl.

Replace the placeholders and run the following command:

```
kubectl patch -n <namespace-name> deployments/<deployment-name> -p '{"spec": {"template": {"metadata": {"labels": {"sidecar.istio.io/inject": "true"} }}}}'
```

## Results

You've enabled Istio sidecar proxy injection for the specified Deployment. The Deployment and all its Pods are labeled with `istio-injection: enabled`. All the Deployment's Pods are instantly injected with an Istio sidecar proxy.

### 4.3.1.1.2 Istio Version

Learn which version of Istio the Istio module contains and how to enable compatibility mode with the previous minor version of Istio.

The version of Istio depends on the version of the Istio module that you use. If a new version of the Istio module introduces a new version of Istio, an upgrade of the module causes an automatic upgrade of Istio. To learn which version of Istio is installed by the newest version of the Istio module, follow [What's New notes](#).

## Compatibility Mode

Compatibility mode allows you to revert certain changes in Istio's behavior, and it is recommended only when you encounter compatibility issues with the new version of Istio. The Istio module supports compatibility with the previous minor version of Istio. For example, for the version of the Istio module that contains Istio 1.24, you can apply a compatibility version of Istio 1.23. See [Compatibility Versions](#).

### ⚠ Caution

You can use the compatibility mode to retain the behavior of the current Istio version before a new version of the Istio module with a higher version of Istio is released. Then, the compatibility is first set to a minor version lower than the one you are currently using. If this lower version's behavior is not compatible with your current mesh setup, some configurations may be broken until the new release of the Istio module is rolled out.

To enable compatibility mode, set the `spec.compatibilityMode` field in the Istio CR to true.

When you set `spec.compatibilityMode: true`, the Istio module applies an opinionated subset of Istio `compatibilityVersion` variables. To learn more about the changes that specific compatibility versions revert, follow [What's New notes](#).

### 4.3.1.1.3 Default Istio Configuration

Within the Istio module, Istio Operator provides baseline values for the Istio installation, which you can override in the Istio custom resource (CR).

See the major differences in the configuration of Istio Operator compared to upstream Istio:

- The Istiod (Pilot) and Ingress Gateway components are enabled by default.
- Automatic Istio sidecar proxy injection is disabled by default.
- To enhance security and performance, both [Istio control plane and data plane](#) use the distroless version of Istio images. Those images are not Debian-based and are slimmed down to reduce any potential attack surface. To learn more, see [Harden Docker Container Images](#).
- Resource requests and limits for Istio sidecars proxies are modified to best suit the needs of the evaluation and production profiles.
- [Mutual TLS](#) (mTLS) is enabled in the STRICT mode for workloads in the Istio service mesh.
- Egress traffic is not controlled. All applications deployed in the Kyma cluster can access outside resources without limitations.
- The CNI component, used for the installation of an Istio sidecar, is provided as a DaemonSet. This means that one replica is present on every node of the target cluster.
- The self-signed CA certificate's bit length is set to 4096 instead of the default 2048.

#### Configuration Based on the Cluster Size

The configuration of Istio resources depends on the cluster capabilities. If your cluster has less than 5 total virtual CPU cores or its total memory capacity is less than 10 gigabytes, the default setup for resources and autoscaling is lighter. If your cluster exceeds both of these thresholds, Istio is installed with the higher resource configuration.

Default Resource Configuration for Smaller Clusters

Component	CPU Requests	CPU Limits	Memory Requests	Memory Limits
Proxy	10 m	250 m	32 Mi	254 Mi
Ingress Gateway	10 m	1000 m	32 Mi	1024 Mi
Pilot	50 m	1000 m	128 Mi	1024 Mi
CNI	10 m	250 m	128 Mi	384 Mi

Default Resource Configuration for Larger Clusters

Component	CPU Requests	CPU Limits	Memory Requests	Memory Limits
Proxy	10 m	1000 m	192 Mi	1024 Mi
Ingress Gateway	100 m	2000 m	128 Mi	1024 Mi
Pilot	100 m	4000 m	512 Mi	2 Gi
CNI	100 m	500 m	512 Mi	1024 Mi

Default Autoscaling Configuration for Smaller Clusters

Component	minReplicas	maxReplicas
Ingress Gateway	1	1
Pilot	1	1

Default Autoscaling Configuration for Larger Clusters

Component	Minimum Number of Replicas	Maximum Number of Replicas
Ingress Gateway	3	10
Pilot	2	5

#### 4.3.1.1.4 Forwarding Client IP in the XFF Header

Learn what the X-Forwarded-For (XFF) header is. Use it to forward client attributes to destination workloads.

### Prerequisites

- You have the Istio module added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- To use CLI instructions, you must install [kubectl](#) and [curl](#). Alternatively, you can use Kyma dashboard.
- You have Istio Ingress Gateway set up. See [Set Up a TLS Gateway in Simple Mode](#).

## Context

To function correctly, many applications must be provided with the client IP address of an originating request. This is often needed in cases where a workload must restrict access based on the client IP address. Reverse proxies pass client attributes to destination workloads using the XFF header.

The XFF header conveys the client IP address and the chain of intermediary proxies that the request traverses to reach the Istio service mesh. The header might not include all IP addresses if an intermediary proxy does not support modifying the header. Due to [technical limitations of AWS Classic ELBs](#), when using an IPv4 connection, the header does not include the public IP of the load balancer in front of Istio Ingress Gateway. Moreover, Istio Ingress Gateway's Envoy does not append the private IP address of the load balancer to the XFF header, effectively removing this information from the request. For more information on XFF, see the [IETF's RFC documentation](#) and [Envoy documentation](#).

To use the XFF header, you must configure the corresponding settings in the Istio custom resource (CR).

## Procedure

- Use Kyma dashboard.
  - a. Go to [Cluster Details](#).
  - b. Choose [Modify Modules](#).
  - c. Select the Istio module.
  - d. Choose [Edit](#).
  - e. In the [General](#) section, add the number of trusted proxies.

Due to the variety of network topologies, the Istio CR must specify the number of trusted proxies deployed in front of the Istio Ingress Gateway proxy in the Istio CR, so that the client address can be extracted correctly.

- f. If you use a Google Cloud or Azure cluster, navigate to the [Gateway](#) section and set the Gateway traffic policy to Local. If you use a different cloud service provider, skip this step.

### ⚠ Caution

For production Deployments, it is strongly recommended to deploy Istio Ingress Gateway Pod to multiple nodes if you enable `externalTrafficPolicy : Local`. For more information, see [Network Load Balancer](#).

Default Istio installation profile configures `PodAntiAffinity` to ensure that Ingress Gateway Pods are evenly spread across all nodes. This guarantees that the above requirement is satisfied if your `IngressGateway` autoscaling configuration `minReplicas` is equal to or greater than the number of nodes. You can configure autoscaling options in the Istio CR using the field `spec.config.components.ingressGateway.k8s.hpaSpec.minReplicas`.

### → Tip

If you use a Google Cloud or Azure cluster, you can find your load balancer's IP address in the field `status.loadBalancer.ingress` of the `ingress-gateway` Service.

- g. Choose **Save**.
- Use kubectl.
  - In the following command, replace the placeholder with the number of trusted proxies deployed in front of the Istio Ingress Gateway proxy. Run the command.

```
kubectl patch istios/default -n kyma-system --type merge -p '{"spec": {"config":{"numTrustedProxies": NUM_OF_TRUSTED_PROXIES}}}'
```

Due to the variety of network topologies, the Istio CR must specify the configuration property `numTrustedProxies`, so that the client IP address can be extracted correctly.

  - If you use a Google Cloud or Azure cluster, run the following command to set the traffic policy to Local. If you use a different cloud service provider, skip this step.

```
kubectl patch istios/default -n kyma-system --type merge -p '{"spec": {"config":{"gatewayExternalTrafficPolicy": "Local"}}}'
```

### Caution

For production Deployments, it is strongly recommended to deploy Istio Ingress Gateway Pod to multiple nodes if you enable `externalTrafficPolicy : Local`. For more information, see [Network Load Balancer](#).

Default Istio installation profile configures `PodAntiAffinity` to ensure that Ingress Gateway Pods are evenly spread across all nodes. This guarantees that the above requirement is satisfied if your `IngressGateway` autoscaling configuration `minReplicas` is equal to or greater than the number of nodes. You can configure autoscaling options in the Istio CR using the field `spec.config.components.ingressGateway.k8s.hpaSpec.minReplicas`.

### Tip

If you use a Google Cloud or Azure cluster, you can find your load balancer's IP address in the field `status.loadBalancer.ingress` of the `ingress-gateway` Service.

## Results

When you call an exposed workload, the response contains the `X-Forwarded-For` and `X-Envoy-External-Address` headers with your public IP address. See an example response for the Client IP 165.1.187.197:

```
{
  "args": {
    "show_env": "true"
  },
  "headers": {
    ...
    "X-Envoy-Attempt-Count": "...",
    "X-Envoy-External-Address": "165.1.187.197",
    "X-Forwarded-Client-Cert": "...",
    "X-Forwarded-For": "165.1.187.197",
    "X-Forwarded-Proto": "...",
    "X-Request-Id": "..."
  },
  "origin": "165.1.187.197",
  "url": "..."}
```

}

→ Tip

You can check your public IP address at <https://api.ipify.org>.

### 4.3.1.1.5 Exposing Workloads Using Gateway API

Use [Gateway API](#) to expose your workload.

#### Prerequisites

- You have the Istio module added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- You have a deployed workload.

⚠ Caution

Exposing an unsecured workload to the outside world is a potential security vulnerability, so be careful. If you want to use this example in a production environment, make sure to secure your workload.

#### Procedure

1. Export the following values as environment variables:

```
export NAMESAPCE={service-namespace}
export BACKENDNAME={service-name}
export PORT={service-port}
```

Option	Description
NAMESAPCE	The name of the backend service.
BACKENDNAME	The name of the backend service that you want to use for routing the incoming HTTP traffic.
PORT	The port number of the backend server to which requests should be forwarded.

2. Install Gateway API CustomResourceDefinitions (CRDs) from the standard channel.

```
kubectl get crd gateways.gateway.networking.k8s.io &> /dev/null || \
{ kubectl kustomize "github.com/kubernetes-sigs/gateway-api/config/crd?
ref=v1.1.0" | kubectl apply -f -; }
```

ⓘ Note

If you've already installed Gateway API CRDs from the experimental channel, you must delete them before installing Gateway API CRDs from the standard channel.

3. Create a Kubernetes Gateway to deploy Istio Ingress Gateway.

```
cat <<EOF | kubectl apply -f -
apiVersion: gateway.networking.k8s.io/v1
kind: Gateway
metadata:
  name: gateway
  namespace: ${NAMESPACE}
spec:
  gatewayClassName: istio
  listeners:
  - name: http
    hostname: "your-domain.kyma.example.com"
    port: 80
    protocol: HTTP
    allowedRoutes:
      namespaces:
        from: Same
EOF
```

This command deploys the Istio Ingress service in your namespace with the corresponding Kubernetes Service of type LoadBalanced and an assigned external IP address.

4. Create an HTTPRoute to configure access to your workload:

```
cat <<EOF | kubectl apply -f -
apiVersion: gateway.networking.k8s.io/v1
kind: HTTPRoute
metadata:
  name: httproute
  namespace: ${NAMESPACE}
spec:
  parentRefs:
  - name: gateway
  hostnames: ["your-domain.kyma.example.com"]
  rules:
  - matches:
    - path:
      type: PathPrefix
      value: /headers
    backendRefs:
    - name: ${BACKENDNAME}
      namespace: ${NAMESPACE}
      port: ${PORT}
EOF
```

## Results

You've exposed your workload. To access it, follow the steps:

1. Discover Istio Ingress Gateway's IP and port.

```
export INGRESS_HOST=$(kubectl get gtw gateway -n ${NAMESPACE} -o
jsonpath='{.status.addresses[0].value}')
export INGRESS_PORT=$(kubectl get gtw gateway -n ${NAMESPACE} -o
jsonpath='{.spec.listeners[?(@.name=="http")].port}')
```

2. Call the service.

```
curl -s -I -HHost:your-domain.kyma.example.com "http://
$INGRESS_HOST:$INGRESS_PORT/headers"
```

### ⓘ Note

This task assumes there's no DNS setup for the `httpbin.kyma.example.com` host, so the call contains the host header.

## 4.3.1.1.6 Exposing Workloads Using oauth2-proxy

Learn how to configure [oauth2-proxy](#) external authorization provider in the Istio custom resource (CR).

### Prerequisites

- You have the Istio module added. If you use a Kyma domain to expose a workload, also the API Gateway module must be added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- You have installed [Helm](#).
- You have a deployed workload.

### Context

The Istio CR allows configuring external authorization providers that operate over HTTP. To set up the oauth2-proxy external authorization provider in the Istio CR and expose your workload using an Istio VirtualService, follow these steps.

### Procedure

1. Export the following values as environment variables:

```
export WORKLOAD_NAME={WORKLOAD_NAME}
export NAMESPACE={WORKLOAD_NAMESPACE}
export DOMAIN_TO_EXPOSE_WORKLOADS={YOUR_DOMAIN}
export GATEWAY={YOUR_GATEWAY}
```

Option	Description
<code>WORKLOAD_NAME</code>	The name of your workload.
<code>NAMESPACE</code>	The name of your workload's namespace.
<code>DOMAIN_TO_EXPOSE_WORKLOADS</code>	The domain that should be used to expose the workloads.
<code>GATEWAY</code>	The Gateway that the VirtualService should use to route traffic to the workload.

2. Create a VirtualService to expose the workload:

```
cat <<EOF | kubectl apply -f -
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: ext-authz
  namespace: $NAMESPACE
spec:
  hosts:
    - "$WORKLOAD_NAME.$DOMAIN_TO_EXPOSE_WORKLOADS"
  gateways:
    - $GATEWAY
  http:
    - match:
        - uri:
            prefix: /
      route:
        - destination:
            port:
              number: 80
            host: $WORKLOAD_NAME.$NAMESPACE.svc.cluster.local
EOF
```

3. Export the following configuration values as environment variables:

```
export CLIENT_ID={YOUR_CLIENT_ID}
export CLIENT_SECRET={YOUR_CLIENT_SECRET}
export COOKIE_SECRET={YOUR_COOKIE_SECRET}
export OIDC_ISSUER_URL={YOUR_OIDC_ISSUER_URL}
```

Option	Description
CLIENT_ID	The unique identifier for the client application that is registered with the external authorizer.
CLIENT_SECRET	A secret key known only to the client and the external authorizer. It is used to authenticate the client when communicating with the external authorizer. To generate this value, you can use the command <code>openssl rand -base64 32   head -c 32   base64</code> .
CLIENT_COOKIE	A secret key used to sign and encrypt the cookies that are used for session management and user authentication. To generate this value, you can use the command <code>openssl rand -base64 32   head -c 32   base64</code> .
OIDC_ISSUER_URL	This is the URL of the OpenID Connect (OIDC) issuer. Typically, you can find the issuer at <a href="https://{{YOUR_IDENTITY_PROVIDER_INSTANCE}}/.well-known/openid-configuration">https://{{YOUR_IDENTITY_PROVIDER_INSTANCE}}/.well-known/openid-configuration</a> .

4. Create a `values.yaml` file with the oauth2-proxy configuration for your authorization server:

→ Tip

You can adapt this configuration to better suit your needs. See the [additional configuration parameters](#).

```
cat <<EOF > values.yaml
config:
  clientId: $CLIENT_ID
  clientSecret: $CLIENT_SECRET
  cookieSecret: $COOKIE_SECRET
extraArgs:
  provider: oidc
  cookie-secure: false
  cookie-domain: "$DOMAIN_TO_EXPOSE_WORKLOADS"
  cookie-samesite: lax
EOF
```

```

set-xauthrequest: true
whitelist-domain: "*.$DOMAIN_TO_EXPOSE_WORKLOADS:*"
reverse-proxy: true
pass-access-token: true
set-authorization-header: true
pass-authorization-header: true
scope: "openid email"
upstream: static://200
skip-provider-button: true
redirect-url: "https://oauth2-proxy.$DOMAIN_TO_EXPOSE_WORKLOADS/oauth2/
callback"
oidc-issuer-url: https://$OIDC_ISSUER_URL
code-challenge-method: S256
EOF

```

- To install oauth2-proxy with the defined configuration, use [oauth2-proxy helm chart](#):

```

helm repo add oauth2-proxy https://oauth2-proxy.github.io/manifests
helm install custom oauth2-proxy/oauth2-proxy -f values.yaml

```

- Register oauth2-proxy as an authorization provider in the Istio module:

```

cat <<EOF | kubectl apply -f -
apiVersion: operator.kyma-project.io/v1alpha1
kind: Istio
metadata:
  name: default
  namespace: kyma-system
spec:
  config:
    numTrustedProxies: 1
    authorizers:
      - name: "oauth2-proxy"
        service: "custom-oauth2-proxy.$NAMESPACE.svc.cluster.local"
        port: 80
        headers:
          inCheck:
            include: ["authorization", "cookie"]
            add:
              x-auth-request-redirect: "https://%REQ(:authority)%%REQ(x-envoy-
original-path?:path)%"
            toUpstream:
              onAllow: ["authorization", "path", "x-auth-request-user", "x-auth-
request-email", "x-auth-request-access-token"]
            toDownstream:
              onAllow: ["set-cookie"]
              onDeny: ["content-type", "set-cookie"]
EOF

```

- Create an AuthorizationPolicy CR with the CUSTOM action and the oauth2-proxy provider:

```

cat <<EOF | kubectl apply -f -
apiVersion: security.istio.io/v1
kind: AuthorizationPolicy
metadata:
  name: ext-authz
  namespace: test
spec:
  action: CUSTOM
  provider:
    name: oauth2-proxy
  rules:
  - to:
    - operation:
      paths:
      - /headers
  selector:
EOF

```

```
matchLabels:  
    app: httpbin  
EOF
```

8. Create a DestinationRule resource with a traffic policy for the external authorization provider:

```
cat <<EOF | kubectl apply -f -  
apiVersion: networking.istio.io/v1  
kind: DestinationRule  
metadata:  
  name: external-authz-https  
  namespace: istio-system  
spec:  
  host: $OIDC_ISSUER_URL  
  trafficPolicy:  
    tls:  
      mode: SIMPLE  
EOF
```

## Results

When you access the URL of the exposed service, you are redirected to the authorization provider's page.

### 4.3.1.1.7 Configuring Istio Access Logs

Use the Telemetry API to selectively enable the Istio access logs.

#### Prerequisites

- You have the Istio module added. See [Add and Delete a Kyma Module \[page 3012\]](#).
- To use CLI instructions, you must install [kubectl](#) and [curl](#). Alternatively, you can use Kyma dashboard.

#### Context

You can enable [Istio access logs](#) to provide fine-grained details about the access to workloads that are part of the Istio service mesh. This can help indicate the four “golden signals” of monitoring (latency, traffic, errors, and saturation) and troubleshooting anomalies. The Istio setup shipped with the Istio module provides a pre-configured [extension provider](#) for access logs, which configures the Istio proxies to print access logs to stdout using the JSON format. It uses a configuration similar to the following one:

```
extensionProviders:  
  - name: stdout-json  
    envoyFileAccessLog:  
      path: "/dev/stdout"  
    logFormat:
```

```
labels:  
  ...  
  traceparent: "%REQ(TRACEPARENT)%"  
  tracestate: "%REQ(TRACESTATE)%"
```

The [log format](#) is based on the Istio default format enhanced with the attributes relevant for identifying the related trace context conforming to the [w3c-tracecontext](#) protocol. See [Traces \[page 1938\]](#) for details. See [Istio tracing](#) on how to enable trace context propagation with Istio.

#### ⚠ Caution

Enabling access logs may drastically increase logs volume and quickly fill up your log storage.

## Configuring Istio Access Logs for a Namespace

### Procedure

- Use Kyma dashboard
  - a. Go to the namespace for which you want to configure Istio access logs.
  - b. Go to *Istio > Telemetries*.
  - c. Choose *Create*.
  - d. Provide a name, for example, access-config.
  - e. Choose *Create*.
- Use kubectl
  - a. Export the name of the namespace for which you want to configure Istio access logs.

```
export YOUR_NAMESPACE=namespace-name
```

- b. To apply the configuration, run:

```
cat <<EOF | kubectl apply -f -  
apiVersion: telemetry.istio.io/v1  
kind: Telemetry  
metadata:  
  name: access-config  
  namespace: $YOUR_NAMESPACE  
spec:  
  accessLogging:  
    - providers:  
      - name: stdout-json  
EOF
```

- c. To verify that the resource is applied, run:

```
kubectl -n $YOUR_NAMESPACE get telemetries.telemetry.istio.io
```

# Configuring Istio Access Logs for a Selective Workload

To configure label-based selection of workloads, use a [selector](#).

## Procedure

- Use Kyma dashboard
  - a. Go to the namespace of the workload for which you want to configure Istio access logs.
  - b. Go to *Istio > Telemetries*.
  - c. Choose *Create*.
  - d. Switch to the YAML section and paste the following configuration into the editor.

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: {your-namespace}
spec:
  selector:
    matchLabels:
      service.istio.io/canonical-name: {your-label}
  accessLogging:
    - providers:
      - name: stdout-json
```

- e. Replace {your-namespace} with the name of the workload's namespace and {your-label} with the name of the workloads' label.
- f. Choose *Create*.

- Use kubectl

- a. Export the name of the workloads' namespace and the workloads' label as environment variables.

```
export YOUR_NAMESPACE={namespace-name}
export YOUR_LABEL={label}
```

- b. To apply the configuration, run:

```
cat <<EOF | kubectl apply -f -
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: $YOUR_NAMESPACE
spec:
  selector:
    matchLabels:
      service.istio.io/canonical-name: $YOUR_LABEL
  accessLogging:
    - providers:
      - name: stdout-json
EOF
```

- c. To verify that the resource is applied, run:

```
kubectl -n $YOUR_NAMESPACE get telemetries.telemetry.istio.io
```

# Configuring Istio Access Logs for a Selective Gateway

Instead of enabling the access logs for all the individual proxies of the workloads you have, you can enable the logs for the proxy used by the related Istio Ingress Gateway.

## Procedure

- Use Kyma dashboard
  - a. Go to the `istio-system`.
  - b. Go to *Istio > Telemetries*.
  - c. Choose *Create*.
  - d. Switch to the YAML section and paste the following configuration into the editor.

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: istio-system
spec:
  selector:
    matchLabels:
      istio: ingressgateway
  accessLogging:
    - providers:
      - name: stdout-json
```

- e. Choose *Create*.
- Use kubectl
  - a. To apply the configuration, run:

```
cat <<EOF | kubectl apply -f -
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: istio-system
spec:
  selector:
    matchLabels:
      istio: ingressgateway
  accessLogging:
    - providers:
      - name: stdout-json
EOF
```

- b. To verify that the resource is applied, run:

```
kubectl -n istio-system get telemetries.telemetry.istio.io
```

## Configuring Istio Access Logs for the Entire Mesh

Enable access logs for all individual proxies of all your workloads and Istio Ingress Gateways.

### Procedure

- Use Kyma dashboard
  - a. Go to the `istio-system`.
  - b. Go to *Istio > Telemetries*.
  - c. Choose *Create*.
  - d. Switch to the YAML section and paste the following configuration into the editor.

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: istio-system
spec:
  accessLogging:
    - providers:
      - name: stdout-json
```

- e. Select *Create*.
- Use kubectl
  - a. To apply the configuration, run:

```
cat <<EOF | kubectl apply -f -
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: access-config
  namespace: istio-system
spec:
  accessLogging:
    - providers:
      - name: stdout-json
EOF
```

- b. To verify that the resource is applied, run:

```
kubectl -n istio-system get telemetries.telemetry.istio.io
```

### 4.3.1.1.8 Troubleshooting for the Istio Module

Troubleshoot problems related to the Istio module.

[Network Connectivity - Basic Diagnostics \[page 1888\]](#)

[Connection Refused Errors \[page 1888\]](#)

[Istio Sidecar Proxy Injection Issues \[page 1889\]](#)

[Reverting the Istio Module's Deletion \[page 1891\]](#)

[SAP HANA Database Connection Issues \[page 1892\]](#)

## 4.3.1.1.8.1 Network Connectivity - Basic Diagnostics

If you're having trouble with network connectivity and don't know where to begin troubleshooting, follow these steps. The issues may not be directly related to Istio, but could be due to misconfigured Istio resources or other cluster resources.

- Verify the state of the Istio custom resource. If it is in the `Warning` state, check the warning message and conditions.
- Verify that no `NetworkPolicy` resources affect the connectivity by blocking traffic between Pods in the service mesh. To get all `NetworkPolicy` resources, run `kubectl get networkpolicies -A`.
- The configuration of the following kinds of resources might affect the connectivity in the service mesh. Verify that these resources are configured as intended:
  - [DestinationRule](#)
  - [PeerAuthentication](#)
  - [Gateway](#)
  - [AuthorizationPolicy](#)
  - [RequestAuthentication](#)
- To check for potential issues in the Istio configuration and see suggestions on how to fix them, run the command `istioctl analyze -A`.
- To enable the access logs of the Envoy proxy, follow the guide [Envoy Access Logs](#). In the access logs, you can find the field `response_flags`. The response flags DC (Downstream client terminated connection) and UC (Upstream terminated connection) are not within the scope of the Istio module, as they relate to the behavior of the client (DC) or the workload application (UC).

## 4.3.1.1.8.2 Connection Refused Errors

### Symptom

You get either the `Connection reset by peer` response or the `GOAWAY` response when you attempt to establish the connection between a service without a sidecar and a service with a sidecar.

### Cause

By default, mutual TLS (mTLS) is enabled in the service mesh. As a result, every element of the service mesh must have an Istio sidecar with a valid TLS certificate to allow communication.

## Solution

- To add a service without a sidecar to the allowlist and disable mTLS traffic for it, create a [DestinationRule](#) .
- To allow connections between a service without a sidecar and a service with a sidecar, create a [PeerAuthentication](#) resource in the PERMISSIVE mode. See [PeerAuthentication](#) .

### 4.3.1.1.8.3 Istio Sidecar Proxy Injection Issues

#### Symptom

Some Pods don't have an Istio sidecar proxy injected.

#### Cause

By default, the Istio module does not automatically inject an Istio sidecar proxy into any Pods you create. To inject a Pod with an Istio sidecar proxy, you must explicitly enable injection for the Pod's Deployment or for the entire namespace. If you have done this and the sidecar is still not installed, follow the solution steps to identify which settings are preventing the injection.

A Pod is not injected with an Istio sidecar proxy if:

- Istio sidecar proxy injection is disabled at the namespace level
- The `sidecar.istio.io/inject` label on the Pod is set to false
- The Pod's `spec` contains `hostNetwork: true`

## Solution

Find out which Pods do not have Istio sidecar proxy injection enabled and why. You can either inspect Pods across all namespaces, focus on a specific namespace, or verify why a selective Pod is not injected.

#### Check Pods in All Namespaces

1. Download the [sidecar analysis script](#) .
2. Run the script.

```
./sidecar-analysis.sh
```

You get an output similar to this one:

```
See all Pods that are not part of the Istio service mesh:  
Pods in namespaces labeled with "istio-injection=disabled":  
- namespace/Pod  
...
```

```
Pods labeled with "sidecar.istio.io/inject=false" in namespaces labeled
with "istio-injection=enabled":
  - namespace/Pod
  ...
Pods not labeled with "sidecar.istio.io/inject=true" in unlabeled
namespaces:
  - namespace/Pod
  ...
```

3. To learn how to include a Pod into the Istio service mesh, see [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#).

## Check Pods in a Selective Namespace

1. Download the [sidecar analysis script](#).
2. Run the script passing the namespace name as a parameter.

```
./sidecar-analysis.sh {YOUR_NAMESPACE}
```

You get an output similar to this one:

```
In the namespace default, the following Pods are not part of the Istio
service mesh:
  - Pod
  - Pod
  ...
Reason: The namespace default has Istio sidecar proxy injection disabled, so
none of its Pods have been injected with an Istio sidecar proxy.
```

3. To learn how to include a Pod into the Istio service mesh, see [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#).

## Check a Selective Pod

- Use Kyma dashboard.
  1. Go to the Pod's namespace.
  2. Check if Istio sidecar proxy injection is enabled at the namespace level.  
Verify if the **Labels** section contains `istio-injection=disabled` or `istio-injection=enabled`. If Istio sidecar proxy injection is disabled at the namespace level, none of its Pods are injected with an Istio sidecar proxy.
  3. Check if Istio sidecar proxy injection is enabled for the Pod's Deployment.  
In the **Workloads** section, select **Deployments** and choose [Edit](#). In the **UI Form** section, check if the `Enable Sidecar Injection` toggle is switched.
  4. Check if the label `sidecar.istio.io/inject: false` is set on a Pod.  
In the **Workloads** section, select **Pods**. Search for `sidecar.istio.io/inject: false`. If your Pod is displayed on the list, it has the label set.
  5. To learn how to include a Pod into the Istio service mesh, see [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#).
- Use kubectl.
  1. Check if Istio sidecar proxy injection is enabled at the namespace level.  
Run the command:

```
kubectl get namespaces {POD_NAMESPACE} -o
jsonpath='{ .metadata.labels.istio-injection }'
```
  2. Check if Istio sidecar proxy injection is enabled for the Pod's Deployment.

Run the command:

```
kubectl get deployments {POD_DEPLOYMENT} -n {NAMESPACE} -o jsonpath='{ .spec.template.metadata.labels }'
```

3. Check if the label `sidecar.istio.io/inject: false` is set on a Pod.

```
kubectl get pod {POD} -n default -o=jsonpath='{.metadata.labels.sidecar.istio.io/inject}'
```

4. To learn how to include a Pod into the Istio service mesh, see [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#).

#### 4.3.1.1.8.4 Reverting the Istio Module's Deletion

Follow the steps outlined in this troubleshooting guide if you unintentionally deleted the Istio module and want to restore the cluster to its normal state without losing any resources created in the cluster.

#### Symptom

The Istio custom resource (CR) is in the `Warning` state. The condition of type `Ready` is set to `false` with the reason `IstioCustomResourcesDangling`. To verify this, run the command:

```
kubectl get istio default -n kyma-system -o jsonpath='{.status.conditions[0]}'
```

You get an output similar to this:

```
{"lastTransitionTime": "2024-09-26T18:23:00Z", "message": "Istio deletion blocked because of existing Istio custom resources", "reason": "IstioCustomResourcesDangling", "status": "False", "type": "Ready"}
```

#### ⓘ Note

If you intended to delete the Istio module, the symptoms described in this document are expected, and you must clean up the remaining resources yourself. To check which resources are blocking the deletion, see the logs of the `istio-controller-manager` container.

#### Cause

The Istio module wasn't completely removed because related resources still exist in the cluster.

For example, the issue occurs when you delete Istio, but there are still `VirtualService` resources either created by you or installed by another Kyma component or module. In such cases, the hooked finalizer pauses the deletion of Istio until you remove all the related resources. This blocking deletion strategy is intentionally designed and is enabled by default for the Istio module.

## Solution

### Use Kyma dashboard.

1. In the *Cluster Details* section, select *Modify Modules*.
2. Select the Istio module.
3. Choose *Edit*.
4. Switch to the *YAML* section.
5. To remove the finalizers from the Istio custom resource, delete the following lines:

```
finalizers:  
  - istios.operator.kyma-project.io/istio-installation
```

When the finalizers are removed, the Istio module is deleted. All the other resources remain in the cluster.

6. Choose *Save*.
7. Add the Istio module again.

When you re-add the Istio module, its reconciliation is reinitiated. The Istio CR returns to the `Ready` state within a few seconds.

### Use kubectl.

1. To edit the Istio CR, run:

```
kubectl edit istio -n kyma-system default
```

2. To remove the finalizers from the Istio custom resource, delete the following lines:

```
finalizers:  
  - istios.operator.kyma-project.io/istio-installation
```

When the finalizers are removed, the Istio module is deleted. All the other resources remain in the cluster.

3. Save the changes.
4. Add the Istio module again.

When you re-add the Istio module, its reconciliation is reinitiated. The Istio CR returns to the `Ready` state within a few seconds.

## 4.3.1.1.8.5 SAP HANA Database Connection Issues

### Symptom

You're unable to connect an application to a SAP HANA Database instance.

### Cause

The Istio module's default configuration does not restrict outbound traffic. This means that the application should have no issues connecting to a SAP HANA Database instance. If you are experiencing issues, they may

be related to the SAP HANA Database instance or your cluster configuration. To identify the source of the problem, follow the troubleshooting steps.

## Solution

### Connect to the SAP HANA Database Instance from Outside of the Cluster

1. Download SAP HANA Client for your operating system from [SAP Development Tools](#).
2. Unpack the downloaded archive.
3. Install SAP HANA Client.
4. To connect to SAP HANA Database instance, use the following command:

```
hdbsql -n <Hana_DB_instance_adress> -u <Hana_DB_user> -p <Hana_DB_password>
```

For example:

```
hdbsql -n aaa.bbb.ccc.ddd:30015 -u my_user -p mypassword
```

If the connection is successful and you can run queries, the issue is not related to the SAP HANA Database instance.

### Connect to the SAP HANA Database Instance from Inside of the Cluster

1. Build a Docker image with the SAP HANA Client installed. You can use the following Dockerfile:

```
FROM eclipse-temurin:17
WORKDIR /build
COPY client.tar client.tar
RUN tar -xvf client.tar
RUN echo "/usr/local/bin" | ./client/hdbinst
ENTRYPOINT ["sleep", "8000"]
```

2. Download SAP HANA Client for Linux x86 64-bit from [SAP Development Tools](#) and save it as `client.tar` in the same directory as the Dockerfile.
3. To build the image, run the following command:

```
docker buildx build --platform=linux/amd64 -t hdbsql .
```

4. To test your image, run the following command:

```
docker run --entrypoint "hdbsql" hdbsql -v
```

You get an output similar to this example:

```
HDBSQL version 2.20.20.1712178305, the SAP HANA Database interactive terminal.
Copyright 2000-2024 by SAP SE.
```

5. Publish the image to a container registry.
6. Run the image in the Kubernetes cluster:

```
kubectl create deployment hdbsql --image={PUBLISHED_IMAGE_NAME}
```

7. To attach to the Pod and connect to the SAP HANA Database instance, run the following command:

```
hdbsql -n {HANA_DB_INSTANCE_ADDRESS} -u {HANA_DB_USER} -p {HANA_DB_PASSWORD}
```

If the connection is successful and you can execute queries, the issue is not related to the setup of the cluster.

8. Check the connection from a Pod that has the Istio sidecar injected. To do this, create a Deployment in a namespace with Istio sidecar injection enabled.

### 4.3.1.2 SAP BTP Operator Module

Use the SAP BTP Operator module to enable service management and consume SAP BTP services from your Kyma cluster.

#### What Is SAP BTP Operator?

The SAP BTP Operator module provides service management, which allows you to consume [SAP BTP services](#) from your Kyma cluster using Kubernetes-native tools. Within the SAP BTP Operator module, [BTP Manager](#) installs the [SAP BTP service operator](#). The SAP BTP service operator enables provisioning and managing service instances and service bindings of SAP BTP services. Consequently, your Kubernetes-native applications can access and use the services from your cluster.

#### Features

The SAP BTP Operator module provides the following features:

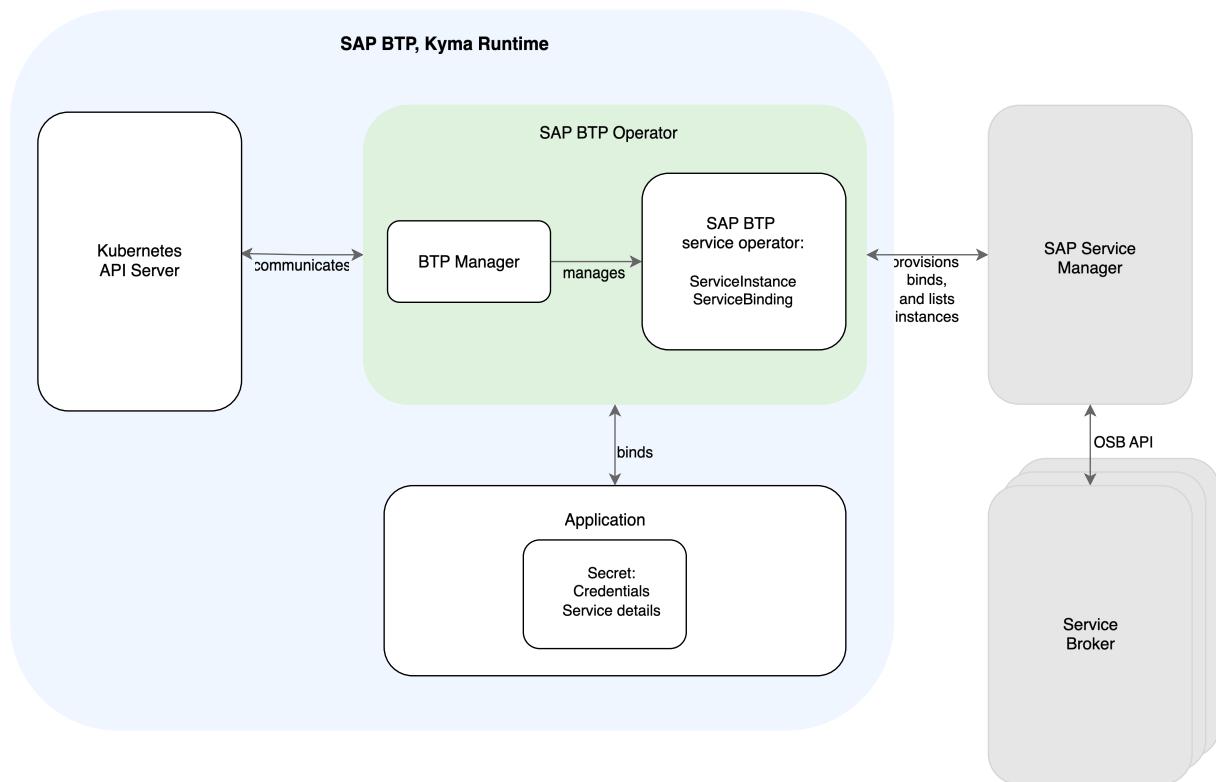
- Credentials and access preconfiguration: Your Secret is readily available on Kyma runtime creation. See [Preconfigured Credentials and Access \[page 1897\]](#).
- Multitenancy: You can configure multiple subaccounts in a single cluster. See [Working with Multiple Subaccounts \[page 1902\]](#).
- Lifecycle management of service instances and service bindings: You can create and delete service instances and service bindings. See [Creating Service Instances and Service Bindings \[page 1899\]](#) and [Deleting Service Bindings and Service Instances \[page 1915\]](#).
- Service binding rotation: You can enhance security by automatically rotating the credentials associated with your service bindings. See [Rotating Service Bindings \[page 1909\]](#).
- Service binding Secret formatting: You can use different attributes in your `ServiceBinding` resource to generate different formats of your Secret resources. See [Formatting Service Binding Secrets \[page 1910\]](#).

#### Scope

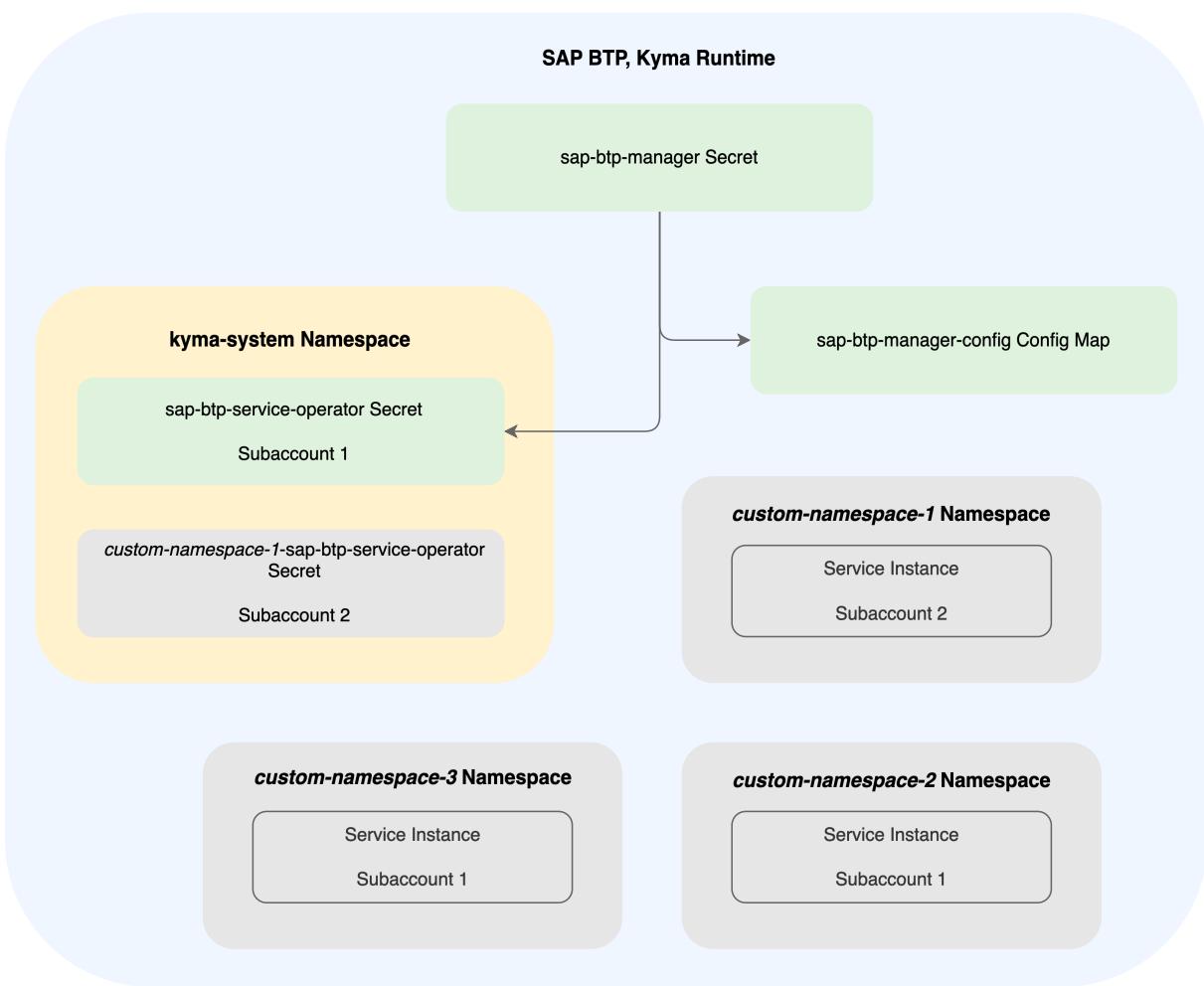
By default, the SAP BTP Operator module consumes SAP BTP services from the subaccount in which you created it. To consume the services from multiple subaccounts in a single Kyma cluster, configure a dedicated Secret for each additional subaccount.

## Architecture

The SAP BTP Operator module provides and retrieves the initial credentials that your application needs to use an SAP BTP service.



Depending on the number of configured Secrets, SAP BTP Operator can have access to multiple subaccounts within your cluster.



**Reference key:** Kyma-Managed Resource User-Managed Resource SAP BTP Operator-Managed Resource

For more information on multitenancy, see [Working with Multiple Subaccounts \[page 1902\]](#).

## SAP BTP, Kyma Runtime

SAP BTP, Kyma runtime runs on a Kubernetes cluster and wraps the SAP BTP Operator module, API server, and one or more applications. The application or multiple applications are the actual consumers of SAP BTP services.

## BTP Manager

BTP Manager is an operator based on the [Kubebuilder](#) framework. It extends Kubernetes API by providing [BtpOperator Custom Resource Definition \(CRD\)](#).

BTP Manager performs the following operations:

- Manages the lifecycle of the SAP BTP service operator, including provisioning of the latest version, updating, and deprovisioning.
- Configures the SAP BTP service operator.

## SAP BTP Service Operator

The SAP BTP service operator is an open-source component that allows you to connect SAP BTP services to your cluster and manage them using Kubernetes-native tools. It is responsible for communicating with

SAP Service Manager. The operator's resources, service instances and service bindings, come from the `services.cloud.sap.com` API group.

## SAP Service Manager

[SAP Service Manager](#) is the central registry for service brokers and platforms in SAP BTP, which enables you to:

- Consume platform services in any connected runtime environment.
- Track the creation and management of service instances.
- Share services and service instances between different runtimes.

SAP Service Manager uses [Open Service Broker API](#) (OSB API) to communicate with service brokers.

## Service Brokers

Service brokers manage the lifecycle of services. SAP Service Manager interacts with service brokers using OSB API to provision and manage service instances and service bindings.

## API/Custom Resource Definitions

The `btpoperators.operator.kyma-project.io` Custom Resource Definition (CRD) describes the kind and the format of data that SAP BTP Operator uses to configure resources.

See the documentation related to the BtpOperator custom resource (CR):

- [SAP BTP Operator Custom Resource](#)
- [Service Instance Custom Resource](#)
- [Service Binding Custom Resource](#)

## Resource Consumption

To learn more about the resources used by the SAP BTP Operator module, see [SAP BTP Operator \[page 3017\]](#).

### 4.3.1.2.1 Preconfigured Credentials and Access

When you create SAP BTP, Kyma runtime, all necessary resources for consuming SAP BTP services are created, and the basic cluster access is configured.

## Credentials

When you create a Kyma instance in the SAP BTP cockpit, the following events happen in your subaccount:

1. An SAP Service Manager service instance with the `service-operator-access` plan is created.
2. An SAP Service Manager service binding with access credentials for the SAP BTP Operator is created.
3. The credentials from the service binding are passed on to the Kyma service instance in the creation process.
4. The `sap-btp-manager` Secret is created and managed in the `kyma-system` namespace.
5. By default, the SAP BTP Operator module is installed together with:
  - The `sap-btp-manager` Secret.
  - The `sap-btp-service-operator` Secret with the access credentials for the SAP BTP service operator. You can view the credentials in the `kyma-system` namespace.
  - The `sap-btp-operator-config` ConfigMap.

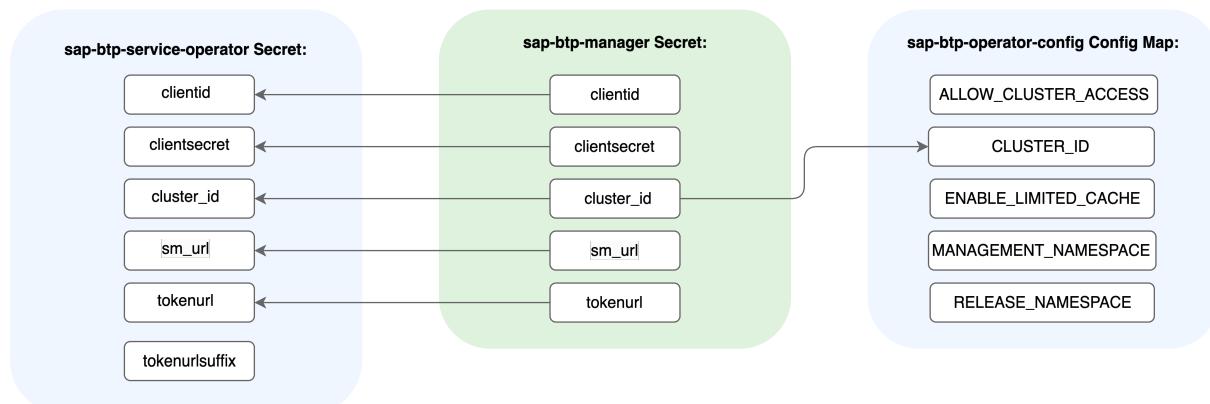
The `sap-btp-manager` Secret provides the following credentials:

- `clientid`
- `clientsecret`
- `cluster_id`
- `sm_url`
- `tokenurl`

### *(i)* Note

If you modify or delete the `sap-btp-manager` Secret, it is modified back to its previous settings or regenerated within up to 24 hours.

The SAP BTP Operator module is added to your cluster by default, and the `sap-btp-manager` Secret generates the SAP BTP service operator's resources as shown in the following diagram:



The cluster ID represents a Kyma service instance created in a particular subaccount and allows for its identification. You can view the cluster ID in Kyma dashboard:

- In the `sap-btp-manager` Secret.
- In the `sap-btp-service-operator` Secret.
- In the `sap-btp-operator-config` ConfigMap.

## Cluster Access

By default, SAP BTP Operator has cluster-wide permissions. You cannot reconfigure the predefined settings. The following parameters manage cluster access:

Cluster Access Parameters

Parameter	Description
CLUSTER_ID	Generated when Kyma runtime is created.
MANAGEMENT_NAMESPACE	Always set to <code>kyma-system</code> .
ALLOW_CLUSTER_ACCESS	You can use every namespace for your operations. The parameter is always set to true. If you change it to false, your setting is automatically reverted.

### 4.3.1.2.2 Creating Service Instances and Service Bindings

To use an SAP BTP service in your Kyma cluster, create its service instance and service binding using Kyma dashboard or kubectl.

#### Prerequisites

- The [SAP BTP Operator module \[page 1894\]](#) is added.  
For instructions on adding modules, see [Add and Delete a Kyma Module \[page 3012\]](#).
- For CLI interactions: `kubectl`  v1.17 or higher.
- You know the service offering name and service plan name for the SAP BTP service you want to connect to your Kyma cluster.  
To find the service and service plan names, in the SAP BTP cockpit, go to  [Services](#)  [Service Marketplace](#). Click on the service tile and find its *name* and *Plan*.

#### Note

You can use [SAP BTP kubectl plugin](#)  to get the available services in your SAP BTP account by using the access credentials stored in the cluster. However, the plugin is still experimental.

## Creating a Service Instance

### Context

To create a service instance, use either Kyma dashboard or kubectl.

## Procedure

- Use Kyma dashboard.
  - a. In the [Namespaces](#) view, go to the namespace you want to work in.
  - b. Go to [Service Management](#) [Service Instances](#).
  - c. Provide the required service details and create a service instance.  
You see the status PROVISIONED.
- Use kubectl.
  - a. To create a ServiceInstance custom resource (CR), follow this example:

```
kubectl create -f - <<EOF
apiVersion: services.cloud.sap.com/v1
kind: ServiceInstance
metadata:
  name: {SERVICE_INSTANCE_NAME}
  namespace: {NAMESPACE}
spec:
  serviceOfferingName: {SERVICE_OFFERING_NAME}
  servicePlanName: {SERVICE_PLAN_NAME}
  externalName: {SERVICE_INSTANCE_NAME}
  parameters:
    key1: val1
    key2: val2
EOF
```

- b. To check the service's status in your cluster, run:

```
kubectl get serviceinstances.services.cloud.sap.com
{SERVICE_INSTANCE_NAME} -n {NAMESPACE}
```

You get an output similar to this one:

NAME	OFFERING
PLAN	STATUS AGE
{SERVICE_INSTANCE_NAME}	{SERVICE_OFFERING_NAME}
{SERVICE_PLAN_NAME}	Created 44s

## Creating a Service Binding

### Context

With a ServiceBinding custom resource (CR), your application can get access credentials for communicating with an SAP BTP service. These access credentials are available to applications through a Secret resource generated in your cluster.

To create a service binding, use either Kyma dashboard or kubectl.

## Procedure

- Use Kyma dashboard.
  - a. In the [Namespaces](#) view, go to the namespace you want to work in.
  - b. Go to [Service Management](#) [Service Bindings](#).
  - c. Choose your service instance name from the dropdown list and create a service binding.  
You see the status PROVISIONED.
- Use kubectl.
  - a. To create a ServiceBinding CR, follow this example:

```
kubectl create -f - <<EOF
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceId: {SERVICE_INSTANCE_NAME}
  externalName: {EXTERNAL_NAME}
  secretName: {SECRET_NAME}
  parameters:
    key1: val1
    key2: val2
EOF
```

### → Remember

In the `serviceInstanceId` field of the service binding, enter the name of the `ServiceInstance` resource you previously created.

- b. To check your service binding status, run:

```
kubectl get servicebindings {BINDING_NAME} -n {NAMESPACE}
```

You see the status `Created`.

- c. Verify the Secret is created with the name specified in the `spec.secretName` field of the `ServiceBinding` CR. The Secret contains access credentials that the applications need to use the service:

```
kubectl get secrets {SECRET_NAME} -n {NAMESPACE}
```

You see the same Secret name as in the `spec.secretName` field of the `ServiceBinding` CR.

## Results

You can use a given service in your Kyma cluster.

### 4.3.1.2.3 Working with Multiple Subaccounts

With the SAP BTP Operator module, you can create configurations for several subaccounts in a single Kyma cluster.

#### Context

By default, a Kyma cluster is associated with one subaccount. Consequently, any service instance created within any namespace is provisioned in the associated subaccount. See [Preconfigured Credentials and Access \[page 1897\]](#). However, with SAP BTP Operator, you can create configurations in a single Kyma cluster that are applied to several subaccounts.

To apply the multitenancy feature, choose the method that suits your needs and application architecture:

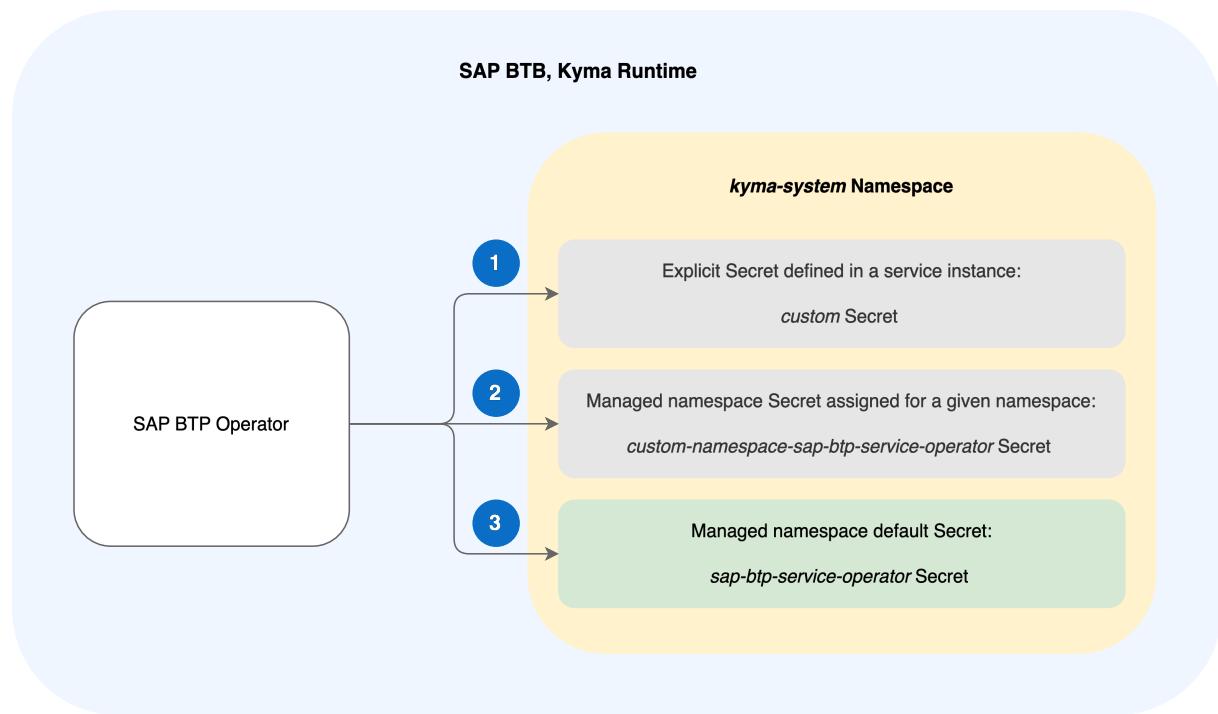
- [Namespace-level mapping \[page 1906\]](#): Connect namespaces to separate subaccounts by configuring dedicated credentials for each namespace.
- [Instance-level mapping \[page 1903\]](#): Define a specific subaccount for each service instance, regardless of the namespace context.

Regardless of the method, you must create Secrets managed in the `kyma-system` namespace.

#### Secrets Precedence

SAP BTP Operator searches for the credentials in the following order:

1. Explicit Secret defined in a service instance
2. Managed namespace Secret assigned for a given namespace
3. Managed namespace default Secret



## Procedure

- To connect a namespace to a specific subaccount, see [Namespace-Level Mapping \[page 1906\]](#).
- To deploy service instances belonging to different subaccounts within the same namespace, see [Instance-Level Mapping \[page 1903\]](#).

### 4.3.1.2.3.1 Instance-Level Mapping

You can map a Kubernetes service instance to an SAP Service Manager instance in a given subaccount. The Service Manager instance is then used to provision that service instance.

## Prerequisites

- A subaccount in the SAP BTP cockpit.
- kubectl configured for communicating with your Kyma instance. See [Access a Kyma Instance Using kubectl \[page 1973\]](#).

## Context

To have multiple service instances from different subaccounts associated with one namespace, you must store access credentials for each subaccount in a custom Secret in the `kyma-system` namespace. To create a service instance with the custom Secret, you must use the `btpAccessCredentialsSecret` field in the `spec` of the service instance. In it, you pass the Secret from the `kyma-system` namespace to create your service instance. You can use different Secrets for different service instances.

## Creating Your Custom Secret

## Procedure

1. In the SAP BTP cockpit, create an SAP Service Manager service instance with the `service-operator-access` plan. See [Creating Instances in Other Environments](#).
2. Create a service binding to the SAP Service Manager service instance you have created. See [Creating Service Bindings in Other Environments](#).
3. Get the access credentials of the SAP Service Manager instance from its service binding. Copy them from the SAP BTP cockpit as a JSON file.
4. Create the `creds.json` file in your working directory and save the credentials there.

5. In the same working directory, generate the Secret by calling the `create-secret-file.sh` script with the `operator` option as the first parameter and `your-secret-name` as the second parameter:

```
curl https://raw.githubusercontent.com/kyma-project/btp-manager/main/hack/create-secret-file.sh | bash -s operator {YOUR_SECRET_NAME}
```

The expected result is the file `btp-access-credentials-secret.yaml` created in your working directory:

```
apiVersion: v1
kind: Secret
type: Opaque
metadata:
  name: {YOUR_SECRET_NAME}
  namespace: kyma-system
data:
  clientid: {CLIENT_ID}
  clientsecret: {CLIENT_SECRET}
  sm_url: {SM_URL}
  tokenurl: {AUTH_URL}
  tokenurlsuffix: "/oauth/token"
```

6. To create the Secret, run:

```
kubectl create -f ./btp-access-credentials-secret.yaml
```

7. To verify if the Secret has been successfully created, run:

```
kubectl get secret -n kyma-system {YOUR_SECRET_NAME}
```

You see the status `Created`.

#### ⓘ Note

You can also view the Secret in Kyma dashboard. In the `kyma-system` namespace, go to [▶ Configuration](#) [▶ Secrets](#), and check the list of Secrets.

## Creating a Service Instance with the Custom Secret

### Context

To create the service instance, use either Kyma dashboard or `kubectl`.

### Procedure

- Use Kyma dashboard.
  - a. In the `Namespaces` view, go to the namespace you want to work in.
  - b. Go to [▶ Service Management](#) [▶ Service Instances](#).
  - c. In the `BTP Access Credentials Secret` field, add the name of the custom Secret you have created.

- d. Provide other required service details and create a service instance.

**⚠ Caution**

Once you set a Secret name in the service instance, you cannot change it in the future.

You see the status PROVISIONED.

- Use kubectl.
  - a. Create your service instance with:
    - The `btpAccessCredentialsSecret` field in the `spec` pointing to the custom pointing to the custom Secret you have created.
    - other parameters as needed

**⚠ Caution**

Once you set a Secret name in the service instance, you cannot change it in the future.

See an example of a `ServiceInstance` CR:

```
kubectl create -f - <<EOF
apiVersion: services.cloud.sap.com/v1
kind: ServiceInstance
metadata:
  name: {SERVICE_INSTANCE_NAME}
  namespace: {NAMESPACE_NAME}
spec:
  serviceOfferingName: {SERVICE_OFFERING_NAME}
  servicePlanName: {SERVICE_PLAN_NAME}
  btpAccessCredentialsSecret: {YOUR_SECRET_NAME}
EOF
```

- b. To verify that your service instance has been created successfully, run:

```
kubectl get serviceinstances.services.cloud.sap.com
{SERVICE_INSTANCE_NAME} -n {NAMESPACE}
```

You see the status `Created` and the message that your service instance has been created successfully. You also see your Secret name in the `btpAccessCredentialsSecret` field of the `spec`.

- c. To verify that you've correctly added the access credentials of the SAP Service Manager instance in your service instance, go to the custom resource (CR) `status` section, and make sure the subaccount ID to which the instance belongs is provided in the `subaccountId` field. The field must not be empty.

## 4.3.1.2.3.2 Namespace-Level Mapping

You can map a Kubernetes namespace to an SAP Service Manager instance in a given subaccount. The Service Manager instance is then used to provision all service instances in that namespace.

### Prerequisites

- A subaccount in the SAP BTP cockpit.
- kubectl configured for communicating with your Kyma instance. See [Access a Kyma Instance Using kubectl \[page 1973\]](#).

### Context

To connect a namespace to a specific subaccount, maintain the access credentials to the subaccount in a Secret dedicated to a specific namespace. Create the {NAMESPACE-NAME}-sap-btp-service-operator Secret in the kyma-system namespace.

## Creating a Namespace-Based Secret

### Procedure

1. In the SAP BTP cockpit, create a new SAP Service Manager service instance with the service-operator-access plan. See [Creating Instances in Other Environments](#).
2. Create a service binding to the SAP Service Manager service instance you have created. See [Creating Service Bindings in Other Environments](#).
3. Get the access credentials of the SAP Service Manager instance from its service binding. Copy them from the SAP BTP cockpit as a JSON file.
4. Create the creds.json file in your working directory and save the credentials there.
5. In the same working directory, call the create-secret-file.sh script with the operator option as the first parameter and namespace-name-sap-btp-service-operator Secret as the second parameter:

```
curl https://raw.githubusercontent.com/kyma-project/btp-manager/main/hack/create-secret-file.sh | bash -s operator {NAMESPACE_NAME}-sap-btp-service-operator
```

The expected result is the btp-access-credentials-secret.yaml file created in your working directory:

```
apiVersion: v1
kind: Secret
type: Opaque
metadata:
```

```

name: {NAMESPACE_NAME}-sap-btp-service-operator
namespace: kyma-system
data:
  clientid: {CLIENT_ID}
  clientsecret: {CLIENT_SECRET}
  sm_url: {SM_URL}
  tokenurl: {AUTH_URL}
  tokenurlsuffix: "/oauth/token"

```

6. To create the Secret, run:

```
kubectl create -f ./btp-access-credentials-secret.yaml
```

You see the status `Created`.

## Creating a Service Instance with a Namespace-Based Secret

### Procedure

1. To create a service instance with a namespace-based Secret, follow the instructions in [Creating Service Instances and Service Bindings \[page 1899\]](#).
2. To verify that you've correctly added the access credentials of the SAP Service Manager instance in your service instance, go to the custom resource (CR) `status` section, and make sure the subaccount ID to which the instance belongs is provided in the `subaccountId` field. The field must not be empty.

### 4.3.1.2.4 Passing Parameters

You can set input parameters for your resources.

### Procedure

To set input parameters, go to the `spec` of the `ServiceInstance` or `ServiceBinding` resource, and use both or one of the following fields:

- `parameters`: Specifies a set of properties sent to the service broker. The specified data is passed to the service broker without any modifications aside from converting it to JSON for transmission to the broker if the `spec` field is specified as YAML. All valid YAML or JSON constructs are supported.

#### Note

You can specify only one parameter field per `spec`.

- `parametersFrom`: Specifies which Secret, together with the key in it, to include in the set of parameters sent to the service broker. The key contains a `string` that represents a JSON file. The `parametersFrom` field is a list that supports multiple sources referenced per `spec`.

If you specified multiple sources in the `parameters` and `parametersFrom` fields, the final payload results from merging all of them at the top level.

If there are any duplicate properties defined at the top level, the specification is invalid. The further processing of the `ServiceInstance` or `ServiceBinding` resource stops with the status `Error`.

## Examples

See the following examples:

- The spec format in YAML:

```
spec:  
  ...  
  parameters:  
    name: value  
  parametersFrom:  
    - secretKeyRef:  
        name: {SECRET_NAME}  
        key: secret-parameter
```

- The spec format in JSON:

```
{  
  "spec": {  
    "parameters": {  
      "name": "value"  
    },  
    "parametersFrom": {  
      "secretKeyRef": {  
        "name": "{SECRET_NAME}",  
        "key": "secret-parameter"  
      }  
    }  
  }  
}
```

- A Secret with the key named `secret-parameter`:

```
apiVersion: v1  
kind: Secret  
metadata:  
  name: {SECRET_NAME}  
type: Opaque  
stringData:  
  secret-parameter:  
    '{  
      "password": "password"  
    }'
```

- The final JSON payload sent to the service broker:

```
{  
  "name": "value",  
  "password": "password"  
}
```

- Multiple parameters in the Secret with key-value pairs separated with commas:

```
secret-parameter:  
  '{  
    "password": "password",  
    "password": "password"  
  }'
```

```
"password": "password",
"key2": "value2",
"key3": "value3"
}'
```

### 4.3.1.2.5 Rotating Service Bindings

Enhance security by automatically rotating the credentials associated with your service bindings. This process involves generating a new service binding while keeping the old credentials active for a specified period to ensure a smooth transition.

#### Enabling Automatic Rotation

To enable automatic service binding rotation, use the `credentialsRotationPolicy` field within the `spec` section of the `ServiceBinding` resource. You can configure the following parameters:

Automatic Rotation Parameters

Parameter	Type	Description	Valid Values
<code>enabled</code>	bool	Turns automatic rotation on or off.	true or false
<code>rotationFrequency</code>	string	Defines the desired interval between binding rotations.	m (minutes) or h (hours)
<code>rotatedBindingTTL</code>	string	Determines how long to keep the old <code>ServiceBinding</code> resource after rotation and before deletion. The actual TTL may be slightly longer.	m (minutes) or h (hours)

#### ⓘ Note

The `credentialsRotationPolicy` does not manage the validity or expiration of the credentials themselves. This is determined by the service you are using.

The `credentialsRotationPolicy` is evaluated periodically during a [control loop](#) on every service binding update or during a complete reconciliation process. This means the actual rotation occurs in the closest upcoming reconciliation loop.

#### Enabling Immediate Rotation

To trigger an immediate rotation regardless of the configured rotation frequency, add the `services.cloud.sap.com/forceRotate: "true"` annotation to the `ServiceBinding` resource. The immediate rotation only works if automatic rotation is already enabled.

The following example shows the configuration of a `ServiceBinding` resource for rotating credentials every 25 days (600 hours) and keeping the old `ServiceBinding` resource for 2 days (48 hours) before deleting it:

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
  credentialsRotationPolicy:
    enabled: true
    rotatedBindingTTL: 48h
    rotationFrequency: 600h
```

## Result

Rotating the service binding has the following results:

- The Secret is updated with the latest credentials.
- The old credentials are kept in a newly-created Secret named `original-secret-name(variable)-guid(variable)`. This temporary Secret is kept until the configured deletion time (TTL). To see the timestamp of the last service binding rotation, go to the `status.lastCredentialsRotationTime` field.

## Limitations

You cannot enable automatic credential rotation for a backup service binding (named: `original-binding-name(variable)-guid(variable)`) marked with the `services.cloud.sap.com/stale` label. This backup service binding is created during the credentials rotation process to facilitate the process.

### 4.3.1.2.6 Formatting Service Binding Secrets

Use different attributes in your `ServiceBinding` resource to generate different formats of your Secret resources.

## Context

Secret resources share a common set of basic parameters that can be divided into two categories:

- Credentials returned from the service broker that allow your application to access and consume an SAP BTP service.
- Attributes of the associated service instance: The details of the service instance itself.

However, the Secret resources can come in various formats:

- Default key-value pairs
- A JSON object
- One JSON object with credentials and service information
- Custom formats

## Key-Value Pairs

If you do not use any of the attributes, the generated Secret is by default in the key-value pair format, as in the following examples:

- Service binding

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
```

- Secret

```
apiVersion: v1
metadata:
  name: {BINDING_NAME}
kind: Secret
data:
  url: {URL}
  client_id: {CLIENT_ID}
  client_secret: {CLIENT_SECRET}
  instance_guid: {SERVICE_INSTANCE_ID}
  instance_name: {SERVICE_INSTANCE_NAME}
  plan: {SERVICE_PLAN_NAME}
  type: {SERVICE_OFFERING_NAME}
```

## Credentials as a JSON Object

To show credentials that the service broker returns within the Secret resource as a JSON object, use the `secretKey` attribute in the service binding `spec`. The value of the `secretKey` is the name of the key that stores the credentials. The credentials are represented in both formats: YAML or JSON.

See the following examples:

- Service binding

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
  secretKey: myCredentials
```

- Secret

```
apiVersion: v1
kind: Secret
metadata:
  name: {BINDING_NAME}
data:
  myCredentials:
    url: {URL}
    client_id: {CLIENT_ID},
    client_secret: {CLIENT_SECRET}
  instance_guid: {SERVICE_INSTANCE_ID}
  instance_name: {SERVICE_INSTANCE_NAME}
  plan: {SERVICE_PLAN_NAME}
  type: {SERVICE_OFFERING_NAME}
```

## Credentials and Service Information as One JSON Object

To show both credentials returned from the service broker and additional service instance attributes as a JSON object, use the `secretRootKey` attribute in the service binding spec.

The `secretRootKey` value is the name of the key that stores both credentials and service instance info. The credentials are represented in both formats: YAML or JSON.

See the following examples:

- Service binding

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
  secretRootKey: myCredentialsAndInstance
```

- Secret

```
apiVersion: v1
kind: Secret
metadata:
  name: {BINDING_NAME}
data:
  myCredentialsAndInstance:
    url: {URL}
    client_id: {CLIENT_ID}
    client_secret: {CLIENT_SECRET}
    instance_guid: {SERVICE_INSTANCE_ID}
    instance_name: {SERVICE_INSTANCE_NAME}
    plan: {SERVICE_PLAN_NAME}
    type: {SERVICE_OFFERING_NAME}
```

## Custom Formats

For additional flexibility, model the Secret resources according to your needs. To generate a custom-formatted Secret, use the `secretTemplate` attribute in the service binding spec. This attribute expects a Go template as its value. For more information, see [Go Templates](#).

Ensure the template is in the YAML format and has the structure of a Kubernetes Secret.

In the provided Secret, you can customize the `metadata` and `data` sections with the following options:

- `metadata`: labels and annotations
- `data`: customize or utilize one of the available formatting options listed in the [Context \[page 1910\]](#) section

### Note

If you customize `data`, it takes precedence over the provided predefined formats.

The provided templates are executed on a map with the following available attributes:

Custom Formatting Parameters

Reference	Description
<code>instance.instance_guid</code>	The service instance ID.
<code>instance.instance_name</code>	The service instance name.
<code>instance.plan</code>	The name of the service plan used to create this service instance.
<code>instance.type</code>	The name of the associated service offering.
<code>credentials.attributes(var)</code>	The content of the credentials depends on a service. For more details, refer to the documentation of the service you're using.
<code>instance.label</code>	The service offering name.

The following examples demonstrate the service binding and generated Secret resources:

- In a service binding with customized `metadata` and `data` sections, you specify both metadata and data in the `secretTemplate`:
  - Service binding

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
  secretTemplate: |
    apiVersion: v1
    kind: Secret
    metadata:
      labels:
        service_plan: {{ .instance.plan }}
    annotations:
      instance: {{ .instance.instance_name }}
  data:
    USERNAME: {{ .credentials.client_id }}
    PASSWORD: {{ .credentials.client_secret }}
```

- Secret

```
apiVersion: v1
kind: Secret
metadata:
  labels:
    service_plan: {SERVICE_PLAN_NAME}
  annotations:
    instance: {SERVICE_INSTANCE_NAME}
data:
  USERNAME: {CLIENT_ID}
  PASSWORD: {CLIENT_SECRET}
```

- In a binding with a customized `metadata` section and applied preexisting formatting option for data with credentials as a JSON object, you omit data from the `secretTemplate` and use the `secretKey` to format your data instead:

- Service binding

```
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
spec:
  serviceInstanceName: {SERVICE_INSTANCE_NAME}
  secretKey: myCredentials
  secretTemplate: |
    apiVersion: v1
    kind: Secret
    metadata:
      labels:
        service_plan: {{ .instance.plan }}
    annotations:
      instance: {{ .instance.instance_name }}
```

- Secret

```
apiVersion: v1
kind: Secret
metadata:
  labels:
    service_plan: {SERVICE_PLAN_NAME}
  annotations:
    instance: {SERVICE_INSTANCE_NAME}
data:
  myCredentials:
    url: {URL}
    client_id: {CLIENT_ID}
    client_secret: {CLIENT_SECRET}
  instance_guid: {SERVICE_INSTANCE_ID}
  instance_name: {SERVICE_INSTANCE_NAME}
  plan: {SERVICE_PLAN_NAME}
  type: {SERVICE_OFFERING_NAME}
```

## 4.3.1.2.7 Deleting Service Bindings and Service Instances

Delete service bindings and service instances using Kyma dashboard or kubectl.

### Context

You can only delete service instances or service bindings created in Kyma using Kyma dashboard or kubectl. You can't perform these operations using the SAP BTP cockpit.

#### ⚠ Caution

Once you delete your service instances and service bindings, you cannot revert the operation.

If you haven't deleted all the service instances and service bindings associated with the `sap-btp-service-operator` Secret in the `kyma-system` namespace, you can't delete your Kyma cluster from the SAP BTP cockpit. To delete the remaining service instances and service bindings, go to Kyma dashboard.

If you have not deleted service instances and bindings connected to your expired free tier service, you can still find the service binding credentials in the SAP Service Manager instance details in the SAP BTP cockpit. Use them to delete the leftover service instances and bindings.

#### → Tip

To successfully delete a service instance, first delete its service bindings.

Use either Kyma dashboard or kubectl to delete a service binding or a service instance.

### Procedure

- Use Kyma dashboard
  - a. In the *Namespaces* view, go to the namespace you want to delete a service binding/instance from.
  - b. Go to *Service Management* and then to *Service Bindings* or *Service Instances*.
  - c. Delete the service binding/instance.
- Use kubectl
  - To delete a service binding, run:

```
kubectl delete servicebindings.services.cloud.sap.com {BINDING_NAME}
```
  - To delete a service instance, run:

```
kubectl delete serviceinstances.services.cloud.sap.com {SERVICE_INSTANCE_NAME}
```

### 4.3.1.3 Application Connector Module

With the Application Connector module, you can connect your workflows deployed on Kyma with external solutions. No matter if you want to integrate an on-premise or a cloud system, the integration process does not change, which prevents any configuration or network-related problems.

#### What is Application Connectivity in Kyma?

Application Connectivity in Kyma simplifies the interaction between external systems and your Kyma workloads. The main benefits are:

- Smooth and loosely coupled integration of external systems with Kyma workloads using [Kyma Eventing](#) ↗
- Easy consumption of SAP BTP services by supporting the SAP BTP Extensibility approach
- Establishing high-security standards for any interaction between systems by using trusted communication channels and authentication methods
- Reducing configuration changes for Kyma workloads through encapsulating configuration details of external API endpoints
- Monitoring and tracing capabilities to facilitate operational aspects

The Application Connector module bundles all features of Application Connectivity in Kyma. You can [install and manage the module using Kyma dashboard \[page 3012\]](#).

The module includes Kubernetes operators and is fully configurable over its own Kubernetes custom resources (CRs). For each external system, a dedicated configuration is used. This allows for individual configuration of security and aspects (like encryption and authentication) per system.

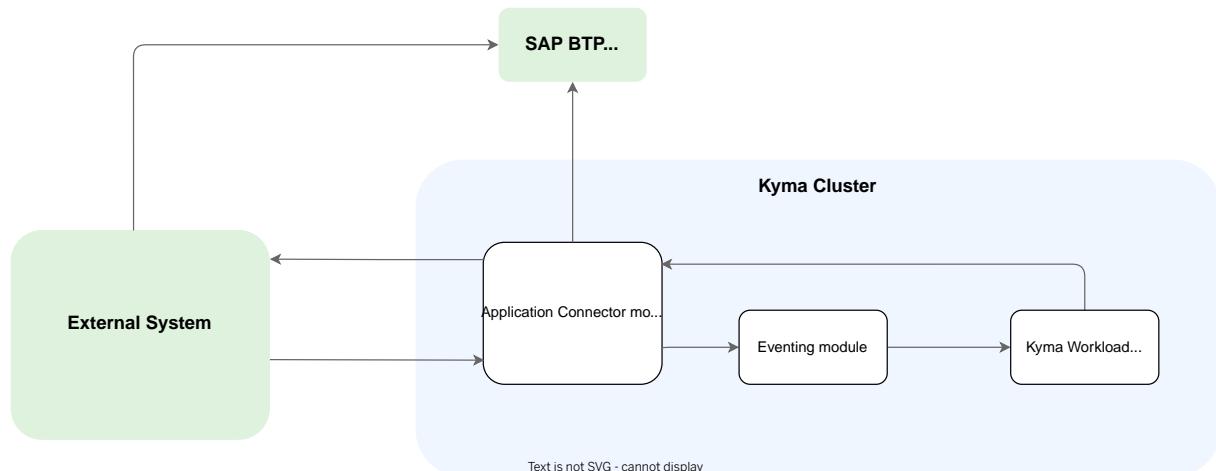
Besides proxying any ingress and egress requests to external systems and dealing with security concerns, it also includes full integration with SAP BTP Unified Customer Landscape (UCL) to simplify the consumption of SAP BTP services.

## Features

The Application Connector module provides the following features:

- Easy installation of Kyma's Application Connectivity capabilities by enabling the Application Connector module in your SAP BTP, Kyma runtime.
- Simple configuration using Kubernetes CRs and easy management with Kyma dashboard.
- Full integration of BTP's UCL service, which implements the SAP BTP Extensibility concept. This allows for the automated integration of external systems registered in the UCL service.
- Dispatching of incoming requests from external systems to Kyma workloads (for example, a [Kyma Serverless Function](#) ↗) by using an Istio Gateway with mTLS and the [Kyma Eventing module](#) ↗ .
- Proxying outgoing requests to external APIs and transparently covering security requirements like encryption and authentication (like OAuth 2.0 + mTLS, Basic Auth, and Client Certificates).
- Metering of throughput and exposing monitoring metrics.

## Architecture



### Application Connector Manager

When you add the Application Connector module, Application Connector Manager takes care of installation and configuration of all the Application Connector module components in your cluster. It manages the lifecycle of the Application Connector module based on the dedicated `ApplicationConnector` custom resource (CRD).

### API/Custom Resource Definitions

The API of the Application Connector module is based on Kubernetes Custom Resource Definitions (CRD), which extend the Kubernetes API with custom additions. To inspect the specification of the Application Connector module API, see:

- [Application CRD](#)
- [CompassConnection CRD](#)
- [ApplicationConnector CRD](#)

### Resource Consumption

To learn more about the resources used by the Application Connector module, see [Application Connector \[page 3016\]](#).

### Useful Links

To learn more how to work with the Application Connector module, see the [Application Connectivity tutorials](#).

To learn more about the architecture, the configuration parameters, and any other references, see [Technical Reference](#) ↗.

#### 4.3.1.4 Keda Module

Learn more about the Keda module. Use it to install and manage the [KEDA](#) ↗ autoscaler in your Kubernetes cluster.

##### What Is KEDA?

Kubernetes-based Event Driven Autoscaler (KEDA) is an autoscaler that allows you to easily scale your Kubernetes-based resources. You can scale your applications on the basis of the data of your choice.

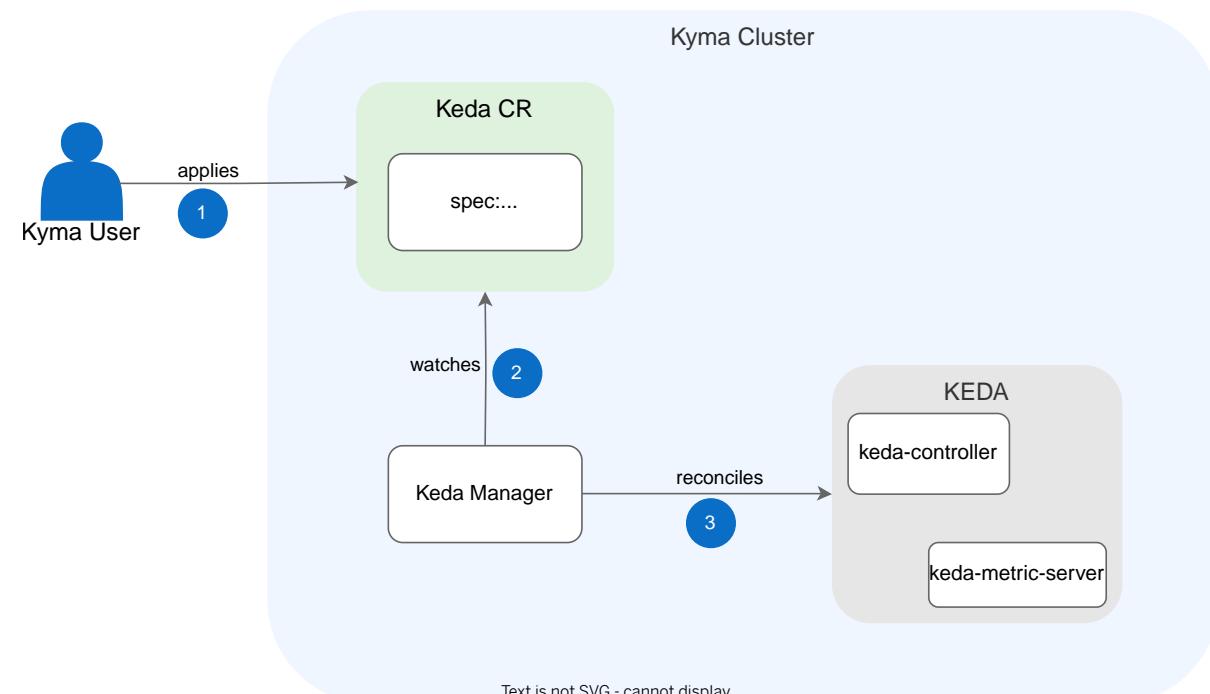
KEDA supports a great number of scalers that help you manage your deployments. For the complete list, check the [KEDA Scalers documentation](#) ↗.

For more information about KEDA features, see the [KEDA documentation](#) ↗.

##### Features

With the Keda module, you can have a custom event-driven autoscaling for Kubernetes workloads.

##### Architecture



1. User applies the Keda custom resource (CR).
2. Keda Manager watches the Keda CR.
3. Keda Manager reconciles the KEDA workloads.

To learn more about the KEDA architecture, see the [KEDA architecture diagram](#).

## Keda Manager

Keda Manager helps you to install and manage KEDA in your cluster. It manages the lifecycle of KEDA based on the dedicated Keda CR.

## API/Custom Resource Definitions

For the Keda CR conditions, check [Keda Custom Resource Conditions \[page 1921\]](#).

To learn more about the KEDA CR, see [KEDA Custom Resources](#).

## Resource Consumption

To learn more about the resources used by the Keda module, see [Keda \[page 3017\]](#).

## Keda Module Demo Applications

To learn how to scale the Kubernetes workloads using the KEDA API based on a simple CPU consumption case and how the Keda module can complement other Kyma components, see [Demo Applications](#).

### 4.3.1.4.1 Keda Module Configuration

Learn how to configure the Keda module using the Keda CustomResourceDefinition (CRD). See how to configure the `logging.level` attribute or the resource consumption.

#### Procedure

- You can change the logging level of the KEDA workloads. To change `logging.level`, choose one of the accepted values:
  - `debug` - is the most detailed option. Useful for a developer during debugging.
  - `info` - provides standard log level indicating operations within the Keda module. For example, it can show whether the workload scaling operation was successful or not.

- `error` - shows error logs only. This means only log messages corresponding to errors and misconfigurations are visible in logs.

```
spec:
  logging:
    operator:
      level: "debug"
```

- To enable the Istio sidecar injection for `operator` and `metricServer`, set the value of `enabledSidecarInjection` to `true`. For example:

```
spec:
  istio:
    metricServer:
      enabledSidecarInjection: true
    operator:
      enabledSidecarInjection: true
```

- To change the resource consumption, enter your preferred values for `operator`, `metricServer`, and `admissionWebhook`. For example:

```
spec:
  resources:
    operator:
      limits:
        cpu: "1"
        memory: "200Mi"
      requests:
        cpu: "150m"
        memory: "150Mi"
    metricServer:
      limits:
        cpu: "1"
        memory: "1000Mi"
      requests:
        cpu: "150m"
        memory: "500Mi"
    admissionWebhook:
      limits:
        cpu: "1"
        memory: "1000Mi"
      requests:
        cpu: "50m"
        memory: "800Mi"
```

## Next Steps

For more information about the Keda resources, see [KEDA concepts](#).

#### 4.3.1.4.2 Keda Custom Resource Conditions

This document describes the possible states of the Keda CR. Two condition types, `Installed` and `Deleted`, are used.

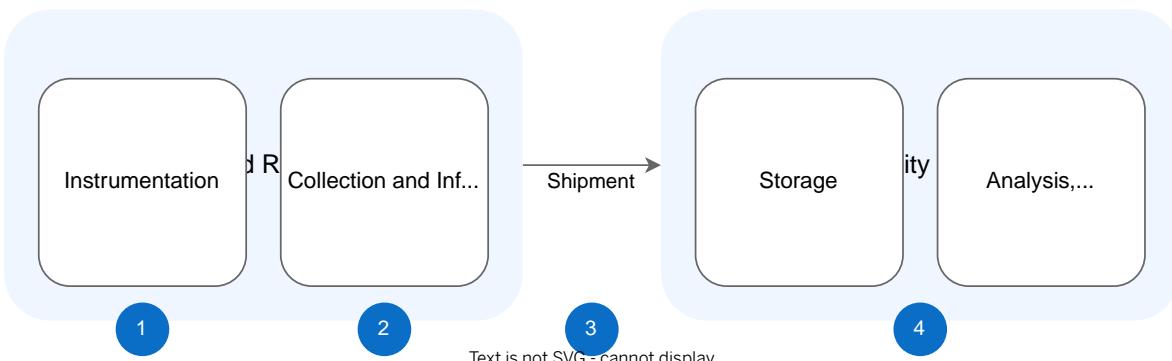
No	CR State	Condition Type	Condition Status	Condition Reason	Remarks
1	Ready	Installed	true	Verified	Server ready
2	Processing	Installed	unknown	Initialized	Initialized
3	Processing	Installed	unknown	Verification	Verification in progress
4	Error	Installed	false	ApplyObjError	Apply object error
5	Error	Installed	false	DeploymentUpdateErr	Deployment update error
6	Error	Installed	false	VerificationErr	Verification error
7	Error	Installed	false	KedaDuplicated	One instance of Keda is allowed
8	Deleted	unknown	Deletion	Deletion in progress	Deletion in progress
9	Deleting	Deleted	true	Deleted	Keda module deleted
10	Error	Deleted	false	DeletionErr	Deletion failed

#### 4.3.1.5 Telemetry Module

Learn more about the Telemetry Module. Use it to enable observability for your application.

##### What Is Telemetry?

Fundamentally, “Observability” is a measure of how well the application’s external outputs can reflect the internal states of single components. The insights that an application and the surrounding infrastructure expose are displayed in the form of metrics, traces, and logs - collectively, that’s called “telemetry” or “signals”. These can be exposed by employing modern instrumentation.



1. In order to implement Day-2 operations for a distributed application running in a container runtime, the single components of an application must expose these signals by employing modern instrumentation.
2. Furthermore, the signals must be collected and enriched with the infrastructural metadata in order to ship them to a target system.
3. Instead of providing a one-size-for-all backend solution, the Telemetry module supports you with instrumenting and shipping your telemetry data in a vendor-neutral way.
4. This way, you can conveniently enable observability for your application by integrating it into your existing or desired backends. Pick your favorite among many observability backends (available either as a service or as a self-manageable solution) that focus on different aspects and scenarios.

The Telemetry module focuses exactly on the aspects of instrumentation, collection, and shipment that happen in the runtime and explicitly defocuses on backends.

### → Tip

An enterprise-grade setup demands a central solution outside the cluster, so we recommend in-cluster solutions only for testing purposes. If you want to install lightweight in-cluster backends for demo or development purposes, see [Integration Guides \[page 1924\]](#).

## Features

To support telemetry for your applications, the Telemetry module provides the following features:

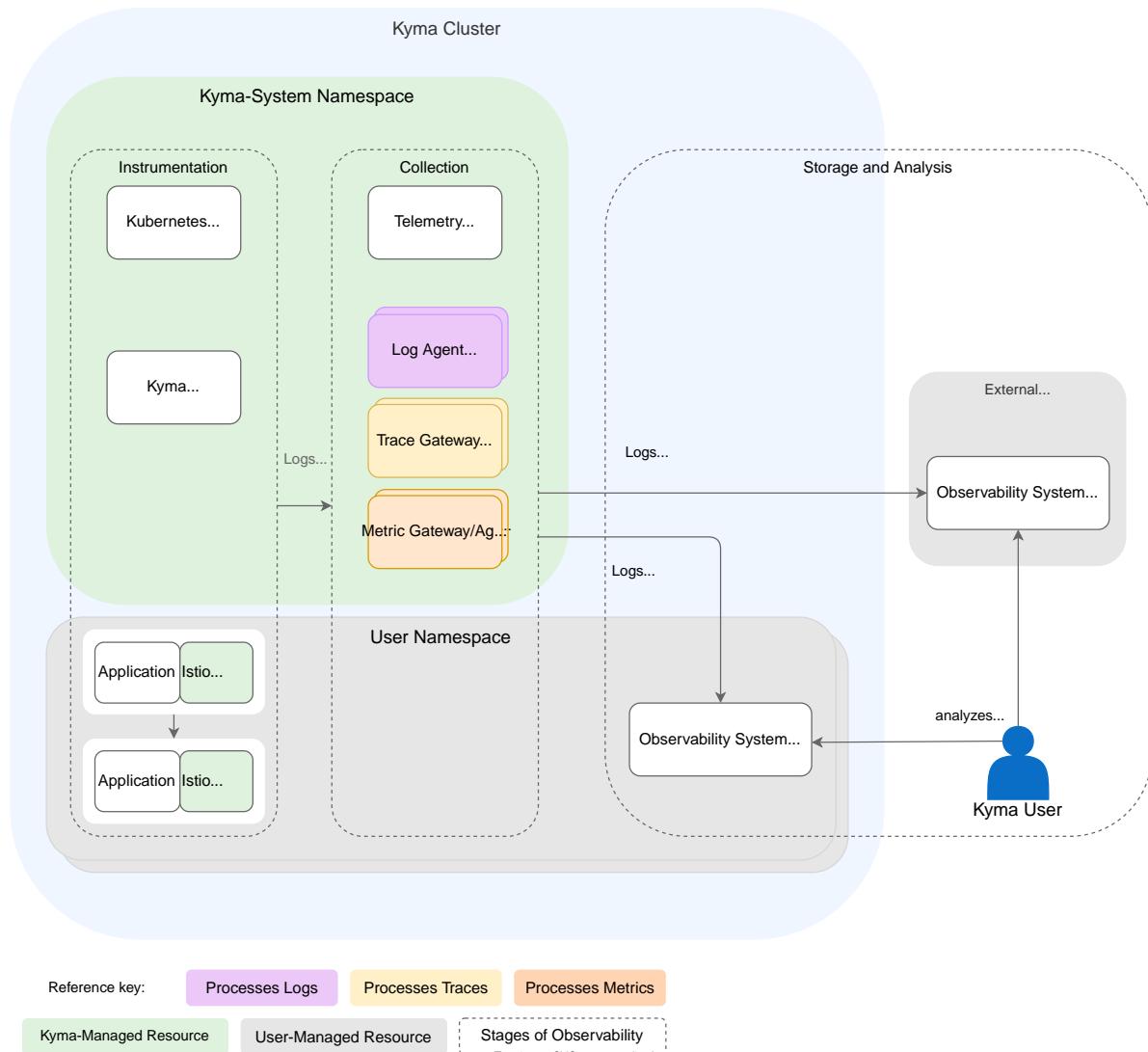
- Tooling for collection, filtering, and shipment: Based on the [Open Telemetry Collector](#) and [Fluent Bit](#), you can configure basic pipelines to filter and ship telemetry data.
- Integration in a vendor-neutral way to a vendor-specific observability system (traces and metrics only): Based on the [OpenTelemetry protocol \(OTLP\)](#), you can integrate backend systems.
- Guidance for the instrumentation (traces and metrics only): Based on [Open Telemetry](#), you get community samples on how to instrument your code using the [Open Telemetry SDKs](#) in nearly every programming language.
- Enriching telemetry data by automatically adding common attributes. This is done in compliance with established semantic conventions, ensuring that the enriched data adheres to industry best practices and is more meaningful for analysis. For details, see [Data Enrichment \[page 1927\]](#).
- Opt-out of features for advanced scenarios: At any time, you can opt out for each data type, and use custom tooling to collect and ship the telemetry data.
- SAP BTP as first-class integration: Integration into SAP BTP Observability services, such as SAP Cloud Logging, is prioritized. For more information, see [Integrate with SAP Cloud Logging \[page 1967\]](#).

## Scope

The Telemetry module focuses only on the signals of application logs, distributed traces, and metrics. Other kinds of signals are not considered. Also, audit logs are not in scope.

Supported integration scenarios are neutral to the vendor of the target system.

## Architecture



### Telemetry Manager

The Telemetry module ships Telemetry Manager as its core component. Telemetry Manager is a Kubernetes operator that implements the Kubernetes controller pattern and manages the whole lifecycle of all other components covered in the Telemetry module. Telemetry Manager watches for the user-created Kubernetes resources: LogPipeline, TracePipeline, and MetricPipeline. In these resources, you specify what data of a signal type to collect and where to ship it. If Telemetry Manager detects a configuration, it deploys the related gateway and agent components accordingly and keeps them in sync with the requested pipeline definition.

For more information, see [Telemetry Manager \[page 1925\]](#).

## Gateways

The Traces and Metrics features share the common approach of providing a gateway based on the [OTel Collector](#). It acts as a central point in the cluster to push data in the OTLP format. From here, the data is enriched and filtered and then dispatched as defined in the individual pipeline resources.

For more information, see [Telemetry Gateways \[page 1927\]](#).

## Log Agent

The log agent is based on a [Fluent Bit](#) installation running as a [DaemonSet](#). It reads all containers' logs in the runtime and ships them according to a [LogPipeline](#) configuration.

For more information, see [Application Logs \[page 1929\]](#).

## Trace Gateway

The trace gateway is based on an [OTel Collector Deployment](#). The gateway provides an [OTLP](#)-based endpoint to which applications can push the trace signals. According to a [TracePipeline](#) configuration, the gateway processes and ships the trace data to a target system.

For more information, see [Traces \[page 1938\]](#) and [Telemetry Gateways \[page 1927\]](#).

## Metric Gateway/Agent

The metric gateway and agent are based on an [OTel Collector Deployment](#) and a [DaemonSet](#). The gateway provides an [OTLP](#)-based endpoint to which applications can push the metric signals. The agent scrapes annotated Prometheus-based workloads. According to a [MetricPipeline](#) configuration, the gateway processes and ships the metric data to a target system.

For more information, see [Metrics \[page 1950\]](#) and [Telemetry Gateways \[page 1927\]](#).

## Integration Guides

To learn about integration with SAP Cloud Logging, read [Integrate with SAP Cloud Logging \[page 1967\]](#).

For integration with other backends, such as Dynatrace, or to learn how to collect data from applications based on the OpenTelemetry Demo App, see [Integration Guides](#).

## API / Custom Resource Definitions

The API of the Telemetry module is based on Kubernetes Custom Resource Definitions (CRD), which extend the Kubernetes API with custom additions. To inspect the specification of the Telemetry module API, see:

- [Telemetry CRD](#)
- [LogPipeline CRD](#)
- [TracePipeline CRD](#)
- [MetricPipeline CRD](#)

## Resource Consumption

To learn more about the resources used by the Telemetry module, see [Telemetry \[page 3017\]](#).

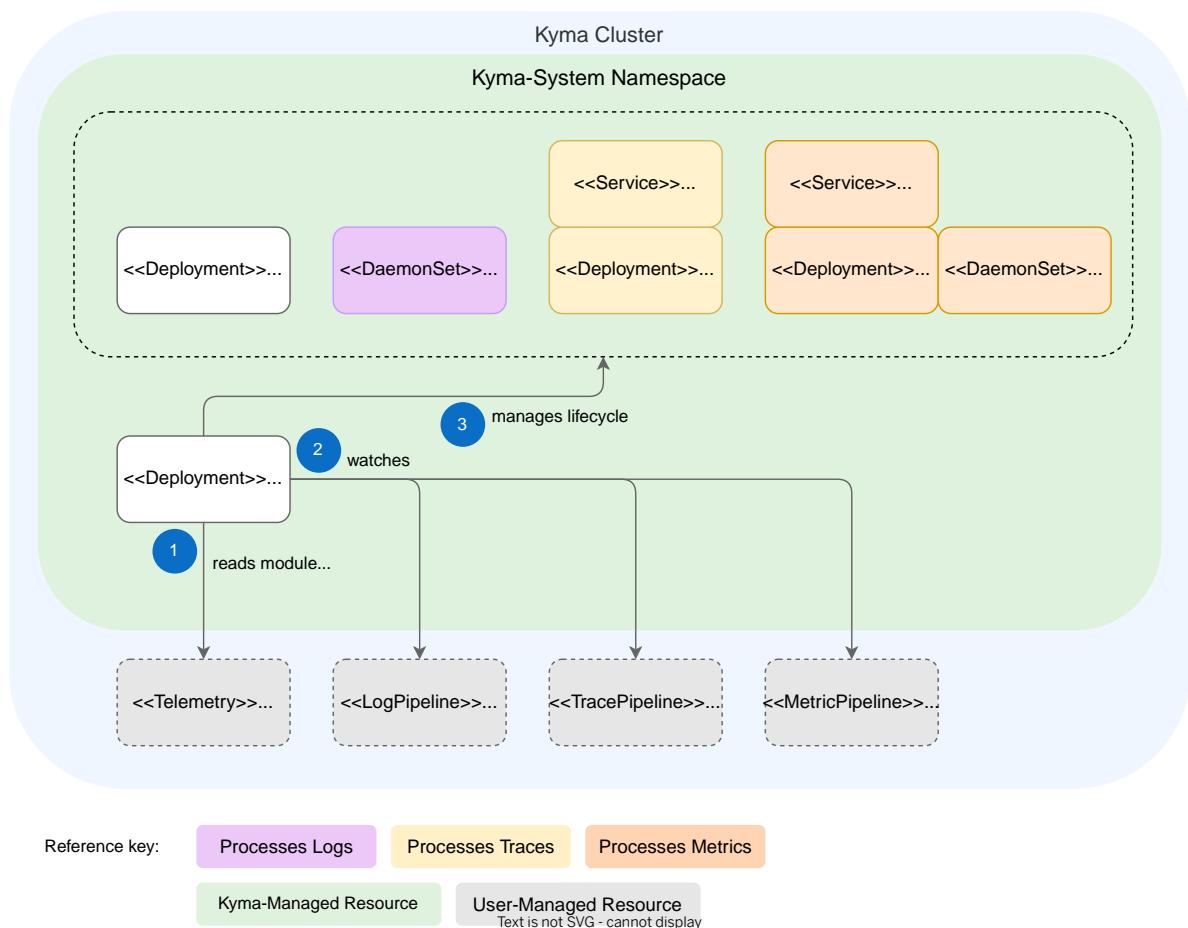
### 4.3.1.5.1 Telemetry Manager

As the core element of the Telemetry module, Telemetry Manager manages the lifecycle of other Telemetry module components by watching user-created resources.

#### Module Lifecycle

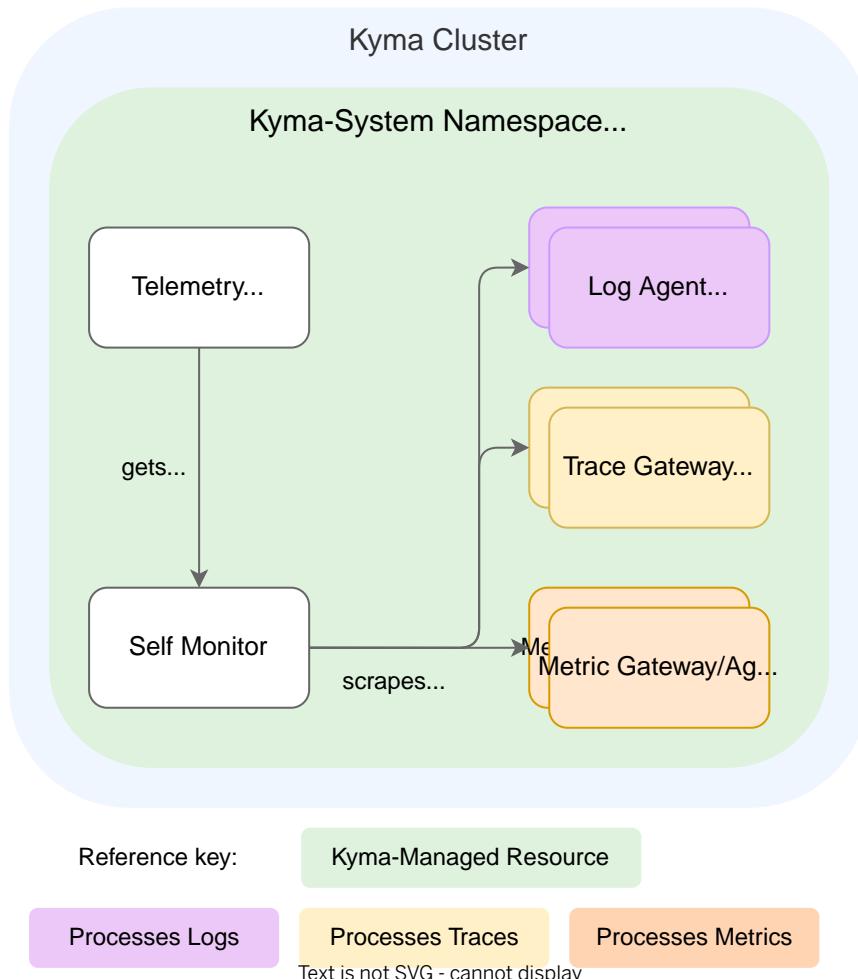
The Telemetry module includes Telemetry Manager, a Kubernetes operator that's described by a custom resource of type `Telemetry`. Telemetry Manager has the following tasks:

1. Watch the module configuration for changes and sync the module status to it.
2. Watch for the user-created Kubernetes resources `LogPipeline`, `TracePipeline`, and `MetricPipeline`. In these resources, you specify what data of a signal type to collect and where to ship it.
3. Manage the lifecycle of the self monitor and the user-configured agents and gateways. For example, only if you defined a `LogPipeline` resource, the Fluent Bit DaemonSet is deployed as log agent.



## Self Monitor

The Telemetry module contains a self monitor, based on [Prometheus](#), to collect and evaluate metrics from the managed gateways and agents. Telemetry Manager retrieves the current pipeline health from the self monitor and adjusts the status of the pipeline resources and the module status.



## Module Configuration and Status

For configuration options and the overall status of the module, see the specification of the related [Telemetry resource](#).

## 4.3.1.5.2 Telemetry Gateways

The Telemetry gateways in Kyma take care of data enrichment, filtering, and dispatching, as well as native support for Istio communication.

### Features

Both, the traces and the metrics feature, are based on a gateway, which is provisioned as soon as you define any pipeline resource. All telemetry data of the related domain passes the gateway, so it acts as a central point and provides the following benefits:

- [Data Enrichment \[page 1927\]](#) to achieve a certain data quality
- Filtering to apply namespace filtering and remove noisy system data (individually for logs, traces, and metrics)
- Dispatching to the configured backends (individually for logs, traces, and metrics)

When the Istio module is added to your Kyma cluster, the gateways support mTLS for the communication from the workload to the gateway, as well as for communication to backends running in the cluster. For details, see [Istio Support \[page 1928\]](#).

The gateways are based on the [OTel Collector](#) and come with a concept of pipelines consisting of receivers, processors, and exporters, with which you can flexibly plug pipelines together (see [Configuration](#)). Kyma's MetricPipeline provides a hardened setup of an OTEL Collector and also abstracts the underlying pipeline concept. Such abstraction has the following benefits:

- Compatibility: An abstraction layer supports compatibility when underlying features change.
- Migratability: Smooth migration experiences when switching underlying technologies or architectures.
- Native Kubernetes support: API provided by Kyma supports an easy integration with Secrets, for example, served by the [SAP BTP Service Operator](#). Telemetry Manager takes care of the full lifecycle.
- Focus: The user doesn't need to understand underlying concepts.

The Telemetry module focuses on full configurability of backends integrated by OTLP. If you need more features than provided by the Kyma MetricPipeline, bring your own collector setup.

### Data Enrichment

The Telemetry gateways automatically enrich your data by adding the following attributes:

- **service.name**: The logical name of the service that emits the telemetry data. The gateway ensures that this attribute always has a valid value.

If not provided by the user, or if its value follows the pattern

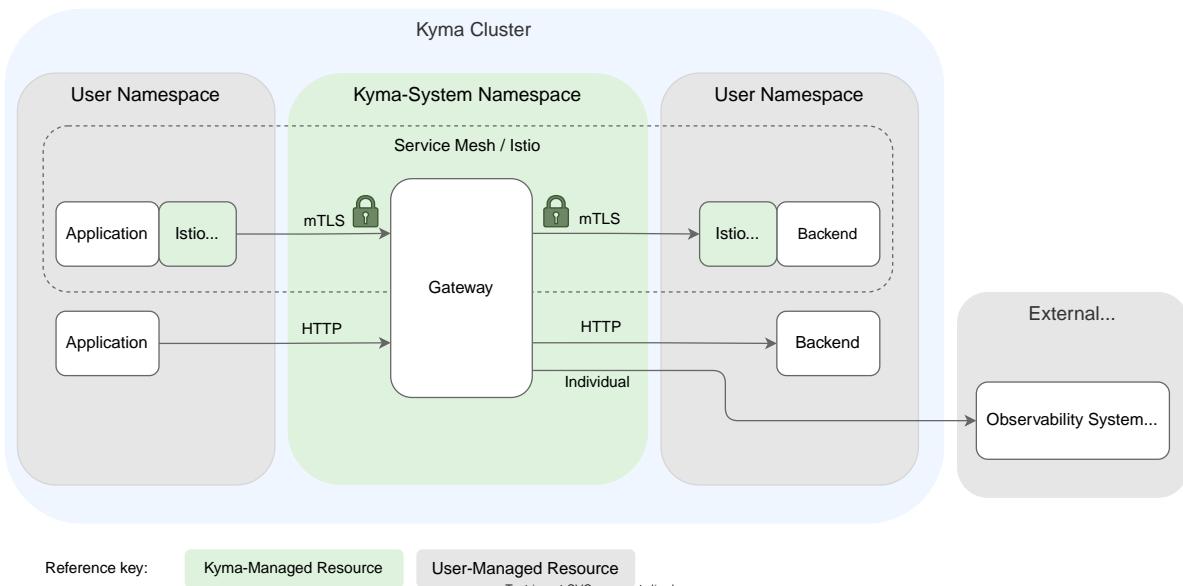
`unknown_service:<process.executable.name>` as described in the [specification](#), then it is generated from Kubernetes metadata. The gateway determines the service name based on the following hierarchy of labels and names:

1. `app.kubernetes.io/name`: Pod label value
2. `app`: Pod label value

3. Deployment/DaemonSet/StatefulSet/Job name
  4. Pod name
  5. If none of the above is available, the value is `unknown_service`.
- **k8s.\*** attributes: These attributes encapsulate various pieces of Kubernetes metadata associated with the Pod, including, but not limited, to:
    - Pod name
    - Deployment/DaemonSet/StatefulSet/Job name
    - Namespace
    - Cluster name

## Istio Support

The Telemetry module automatically detects whether the Istio module is added to your cluster, and injects Istio sidecars to the Telemetry components. Additionally, the ingestion endpoints of gateways are configured to allow traffic in the permissive mode, so they accept mTLS-based communication as well as plain text.



Clients in the Istio service mesh transparently communicate to the gateway with mTLS. Clients that don't use Istio can communicate with the gateway in plain text mode. The same pattern applies for the communication to the backends running in the cluster. External clusters use the configuration as specified in the pipelines output section.

### 4.3.1.5.3 Application Logs

With application logs, you can debug an application and derive the internal state of an application. When logs are emitted with the correct severity level and context, they're essential for observing an application.

#### Overview

The Telemetry module provides the [Fluent Bit](#) log agent for the collection and shipment of application logs of any container running in the Kyma runtime.

You can configure the log agent with external systems using runtime configuration with a dedicated Kubernetes API ([CRD](#)) named `LogPipeline`. With the `LogPipeline`'s HTTP output, you can natively integrate with vendors that support this output, or with any vendor using a [Fluentd integration](#).

The feature is optional, if you don't want to use the Logs feature, simply don't set up a `LogPipeline`.

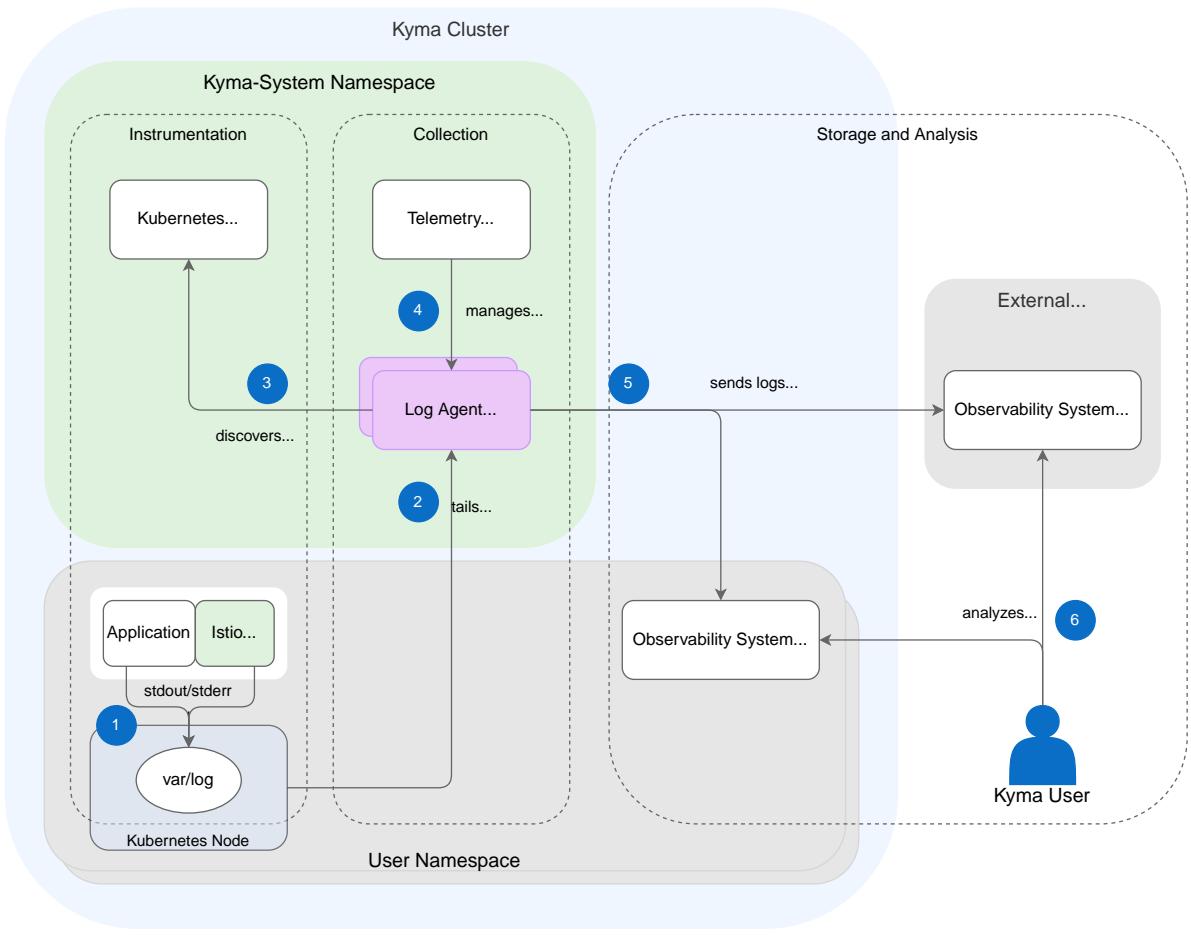
#### Prerequisites

Your application must log to `stdout` or `stderr`, which ensures that the logs can be processed by Kubernetes primitives like `kubectl logs`. For details, see [Kubernetes: Logging Architecture](#).

#### Architecture

##### Log Agent

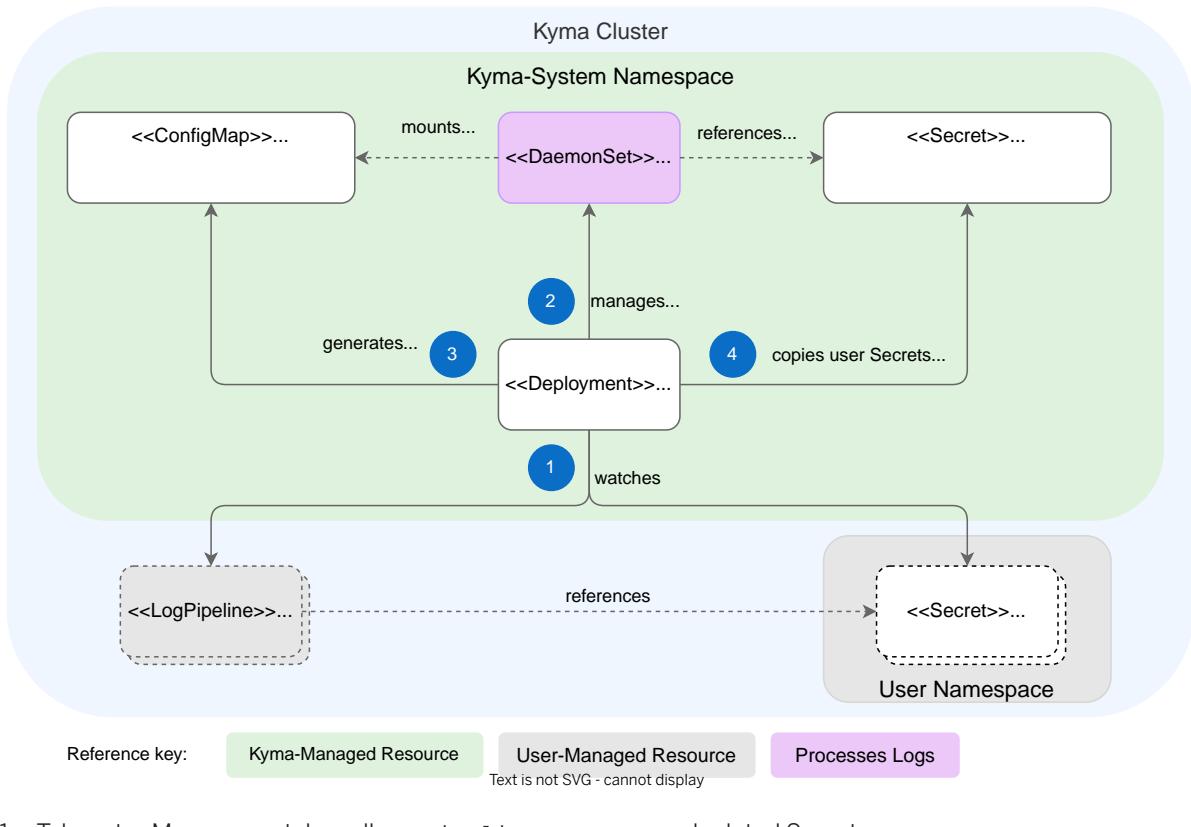
In the Kyma cluster, the Telemetry module provides a DaemonSet of [Fluent Bit](#) acting as a agent. The agent tails container logs from the Kubernetes container runtime and ships them to a backend.



1. Container logs are stored by the Kubernetes container runtime under the `var/log` directory and its subdirectories.
2. Fluent Bit runs as a [DaemonSet](#) (one instance per Node), detects any new log files in the folder, and tails them using a filesystem buffer for reliability.
3. Fluent Bit discovers additional Pod metadata, such as Pod annotations and labels.
4. Telemetry Manager configures Fluent Bit with your output configuration, observes the log flow, and reports problems in the `LogPipeline` status.
5. The log agent sends the data to the observability system that's specified in your `LogPipeline` resource - either within the Kyma cluster, or, if authentication is set up, to an external observability backend. You can use the integration with HTTP to integrate a system directly or with an additional Fluentd installation.
6. To analyze and visualize your logs, access the internal or external observability system.

## Telemetry Manager

The `LogPipeline` resource is watched by Telemetry Manager, which is responsible for generating the custom parts of the Fluent Bit configuration.



1. Telemetry Manager watches all LogPipeline resources and related Secrets.
2. Furthermore, Telemetry Manager takes care of the full lifecycle of the Fluent Bit DaemonSet itself. Only if you defined a LogPipeline, the agent is deployed.
3. Whenever the configuration changes, Telemetry Manager validates the configuration (with a [validating webhook](#)) and generates a new configuration for the Fluent Bit DaemonSet, where several ConfigMaps for the different aspects of the configuration are generated.
4. Referenced Secrets are copied into one Secret that is also mounted to the DaemonSet.

## Log Agent

If a LogPipeline is defined, a DaemonSet is deployed acting as an agent. The agent is based on [Fluent Bit](#) and encompasses the collection of application logs provided by the Kubernetes container runtime. The agent sends all data to the configured backend.

## Setting up a LogPipeline

In the following steps, you can see how to construct and deploy a typical LogPipeline. Learn more about the available [parameters and attributes](#).

### 1. Create a LogPipeline and Output

To ship application logs to a new output, create a resource of the kind LogPipeline and save the file (named, for example, logpipeline.yaml).

```
kind: LogPipeline
```

```

apiVersion: telemetry.kyma-project.io/v1alpha1
metadata:
  name: http-backend
spec:
  output:
    http:
      dedot: false
      port: "80"
      uri: "/"
      host:
        value: https://myhost/logs
      user:
        value: "user"
      password:
        value: "not-required"

```

An output is a data destination configured by a [Fluent Bit output](#) of the relevant type. The `LogPipeline` supports the `http` output type: It sends the data to the specified HTTP destination. The output is designed to integrate with a [Fluentd HTTP Input](#), which opens up a huge ecosystem of integration possibilities.

## 2. Filter Your Input

`kube-system`, `istio-system`, `kyma-system`), which are excluded by default.

To filter your application logs by namespace or container, use an `input` spec to restrict or specify which resources you want to include. For example, you can define the namespaces to include in the input collection, exclude namespaces from the input collection, or choose that only system namespaces are included. Learn more about the available By default, input is collected from all namespaces, except the system namespaces ([parameters and attributes](#)).

The following example collects input from all namespaces excluding `kyma-system` and only from the `istio-proxy` containers:

```

kind: LogPipeline
apiVersion: telemetry.kyma-project.io/v1alpha1
metadata:
  name: http-backend
spec:
  input:
    application:
      namespaces:
        exclude:
          - kyma-system
      containers:
        include:
          - istio-proxy
  output:
    ...

```

It might happen that Fluent Bit prints an error per processed log line, which is then collected and re-processed. To avoid problems with such recursive logs, it is recommended that you exclude the logs of the Fluent Bit container. The following example collects input from all namespaces including system namespaces, but excludes the Fluent Bit container:

```

spec:
  input:
    application:
      namespaces:
        system: true
      containers:
        exclude:
          - fluent-bit

```

### 3. Add Authentication Details From Secrets

By defining integrations into external systems usually need authentication details dealing with sensitive data. To handle that data properly in Secrets, LogPipeline supports the reference of Secrets.

Using the http output definition and the valueFrom attribute, you can map Secret keys for mutual TLS (mTLS) or Basic Authentication:

- mTLS:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: LogPipeline
metadata:
  name: http-backend
spec:
  output:
    http:
      dedot: false
      port: "80"
      uri: "/"
      host:
        valueFrom:
          secretKeyRef:
            name: http-backend-credentials
            namespace: default
            key: HTTP_ENDPOINT
    tls:
      cert:
        valueFrom:
          secretKeyRef:
            name: http-backend-credentials
            namespace: default
            key: TLS_CERT
      key:
        valueFrom:
          secretKeyRef:
            name: http-backend-credentials
            namespace: default
            key: TLS_KEY
  input:
  ...
  filters:
  ...
```

- Basic Authentication:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: LogPipeline
metadata:
  name: http-backend
spec:
  output:
    http:
      dedot: false
      port: "80"
      uri: "/"
      host:
        valueFrom:
          secretKeyRef:
            name: http-backend-credentials
            namespace: default
            key: HTTP_ENDPOINT
    user:
      valueFrom:
        secretKeyRef:
          name: http-backend-credentials
          namespace: default
```

```

    key: HTTP_USER
  password:
    valueFrom:
      secretKeyRef:
        name: http-backend-credentials
        namespace: default
        key: HTTP_PASSWORD
  input:
    ...
  filters:
    ...

```

The related Secret must have the referenced name, be located in the referenced namespace, and contain the mapped key. See the following example:

```

kind: Secret
apiVersion: v1
metadata:
  name: http-backend-credentials
stringData:
  HTTP_ENDPOINT: https://myhost/logs
  HTTP_USER: myUser
  HTTP_PASSWORD: XXX
  TLS_CERT: ...
  TLS_KEY: ...

```

## 4. Rotate the Secret

Telemetry Manager continuously watches the Secret referenced with the `secretKeyRef` construct. You can update the Secret's values, and Telemetry Manager detects the changes and applies the new Secret to the setup.

### → Tip

If you use a Secret owned by the [SAP BTP Service Operator](#), you can configure an automated rotation using a `credentialsRotationPolicy` with a specific `rotationFrequency` and don't have to intervene manually.

## 5. Deploy the Pipeline

To activate the `LogPipeline`, apply the `logpipeline.yaml` resource file in your cluster:

```
kubectl apply -f logpipeline.yaml
```

### Result

You activated a `LogPipeline` and logs start streaming to your backend.

To check that the pipeline is running, wait until the status conditions of the `LogPipeline` in your cluster have status `True`:

### ↔ Output Code

```
kubectl get logpipeline
NAME      CONFIGURATION GENERATED   AGENT HEALTHY   FLOW HEALTHY
backend   True                      True          True
```

## Log Record Processing

After a log record has been read, it is preprocessed by configured plugins, like the `kubernetes` filter. Thus, when a record is ready to be processed by the sections defined in the `LogPipeline` definition, it has several attributes available for processing and shipment.



Learn more about the flow of the log record through the general pipeline and the available log attributes in the following stages:

### Container Log Message

The following example assumes that there's a container `myContainer` of Pod `myPod`, running in namespace `myNamespace`, logging to `stdout` with the following log message in the JSON format:

#### ↔ Output Code

```
{
  "level": "warn",
  "message": "This is the actual message",
  "tenant": "myTenant",
  "traceID": "123"
}
```

### Tail Input

The `tail` input plugin reads the log message from a log file managed by the container runtime. The input plugin brings a dedicated filesystem buffer for the pipeline. The file name contains the namespace, Pod, and container information that will be available later as part of the `tag` . The tag is prefixed with the pipeline name. The resulting log record available in an internal Fluent Bit representation looks similar to the following example:

#### ↔ Output Code

```
{
  "time": "2022-05-23T15:04:52.193317532Z",
  "stream": "stdout",
  "_p": "F",
  "log": "{\"level\": \"warn\", \"message\": \"This is the actual message\", \"tenant\": \"myTenant\", \"traceID\": \"123\"}"
}
```

The attributes have the following meaning:

- `time`: The timestamp generated by the container runtime at the moment the log was written to the log file.
- `stream`: The stream to which the application wrote the log, either `stdout` or `stderr`.
- `_p`: Indicates if the log message is partial (P) or final (F). Optional, dependent on container runtime. Because a CRI multiline parser is applied for the tailing phase, all multilines on the container runtime level are aggregated already and no partial entries must be left.

- `log`: The raw and unparsed log message.

## Kubernetes Filter (Metadata)

After the tail input, the [Kubernetes filter](#) is applied. The container information from the log file name (available in the tag) is interpreted and used for a Kubernetes API Server request to resolve more metadata of the container. All the resolved metadata enrich the existing record as a new attribute `kubernetes`:

### ↳ Output Code

```
{
  "kubernetes": {
    "pod_name": "myPod-74db47d99-ppnsw",
    "namespace_name": "myNamespace",
    "pod_id": "88dbd1ef-d977-4636-804d-ef220454be1c",
    "host": "myHost1",
    "container_name": "myContainer",
    "docker_id": "5649c36fcc1e956fc95e3145441f427d05d6e514fa439f4e4f1cce80fb2c037",
    "container_hash": "myImage@sha256:1f8d852989c16345d0e81a7bb49da231ade6b99d51b95c56702d04c417549b26",
    "container_image": "myImage:myImageTag",
    "labels": {
      "app": "myApp",
      "sidecar.istio.io/inject": "true",
      ...
    }
  }
}
```

## Kubernetes Filter (JSON Parser)

After the enrichment of the log record with the Kubernetes-relevant metadata, the [Kubernetes filter](#) also tries to parse the record as a JSON document. If that is successful, all the parsed root attributes of the parsed document are added as new individual root attributes of the log.

The record **before** applying the JSON parser:

### ↳ Output Code

```
{
  "time": "2022-05-23T15:04:52.193317532Z",
  "stream": "stdout",
  "_p": "F",
  "log": "{\"level\": \"warn\", \"message\": \"This is the actual message\", \"tenant\": \"myTenant\", \"traceID\": \"123\"}",
  "kubernetes": {...}
}
```

The record **after** applying the JSON parser:

### ↳ Output Code

```
{
  "time": "2022-05-23T15:04:52.193317532Z",
  "stream": "stdout",
  "_p": "F",
```

```
"log": "{\"level\": \"warn\", \"message\": \"This is the actual message\", \"tenant\": \"myTenant\", \"traceID\": \"123\"},  
  \"kubernetes\": {...},  
  \"level\": \"warn\",  
  \"message\": \"This is the actual message\",  
  \"tenant\": \"myTenant\",  
  \"traceID\": \"123\"\n}
```

## Operations

The Telemetry module ensures that the log agent instances are operational and healthy at any time, for example, with buffering and retries. However, there may be situations when the instances drop logs, or cannot handle the log load.

To detect and fix such situations, check the pipeline status and check out [Troubleshooting \[page 1937\]](#).

## Limitations

- **Reserved Log Attributes:** The log attribute named `kubernetes` is a special attribute that's enriched by the `kubernetes` filter. When you use that attribute as part of your structured log payload, the metadata enriched by the filter are overwritten by the payload data. Filters that rely on the original metadata might no longer work as expected.
- **Buffer Limits:** Fluent Bit buffers up to 1 GB of logs if a configured output cannot receive logs. The oldest logs are dropped when the limit is reached or after 300 retries.
- **Throughput:** Each Fluent Bit Pod (each running on a dedicated Node) can process up to 10 MB/s of logs for a single `LogPipeline`. With multiple pipelines, the throughput per pipeline is reduced. The used logging backend or performance characteristics of the output plugin might limit the throughput earlier.
- **Max Amount of Pipelines:** The maximum amount of `LogPipeline` resources is 5.

## Troubleshooting

### No Logs Arrive at the Backend

#### Symptom:

- No logs arrive at the backend.
- In the `LogPipeline` status, the `TelemetryFlowHealthy` condition has status `AllDataDropped`.

**Cause:** Incorrect backend endpoint configuration (for example, using the wrong authentication credentials) or the backend being unreachable.

#### Solution:

- Check the `telemetry-fluent-bit` Pods for error logs by calling `kubectl logs -n kyma-system {POD_NAME}`.

- Check if the backend is up and reachable.

### Not All Logs Arrive at the Backend

**Symptom:** The backend is reachable and the connection is properly configured, but some logs are refused. In the `LogPipeline` status, the `TelemetryFlowHealthy` condition has status `SomeDataDropped`.

**Cause:** It can happen due to a variety of reasons. For example, a possible reason may be that the backend is limiting the ingestion rate, or the backend is refusing logs because they are too large.

**Solution:**

1. Check the `telemetry-fluent-bit` Pods for error logs by calling `kubectl logs -n kyma-system {POD_NAME}`. Also, check your observability backend to investigate potential causes.
2. If backend is limiting the rate by refusing logs, try the options described in [Agent Buffer Filling Up](#).
3. Otherwise, take the actions appropriate to the cause indicated in the logs.

### Agent Buffer Filling Up

**Symptom:** In the `LogPipeline` status, the `TelemetryFlowHealthy` condition has status `BufferFillingUp`.

**Cause:** The backend export rate is too low compared to the log collection rate.

**Solution:**

- Option 1: Increase maximum backend ingestion rate. For example, by scaling out the SAP Cloud Logging instances.
- Option 2: Reduce emitted logs by re-configuring the `LogPipeline` (for example, by applying namespace or container filters).
- Option 3: Reduce emitted logs in your applications (for example, by changing severity level).

## 4.3.1.5.4 Traces

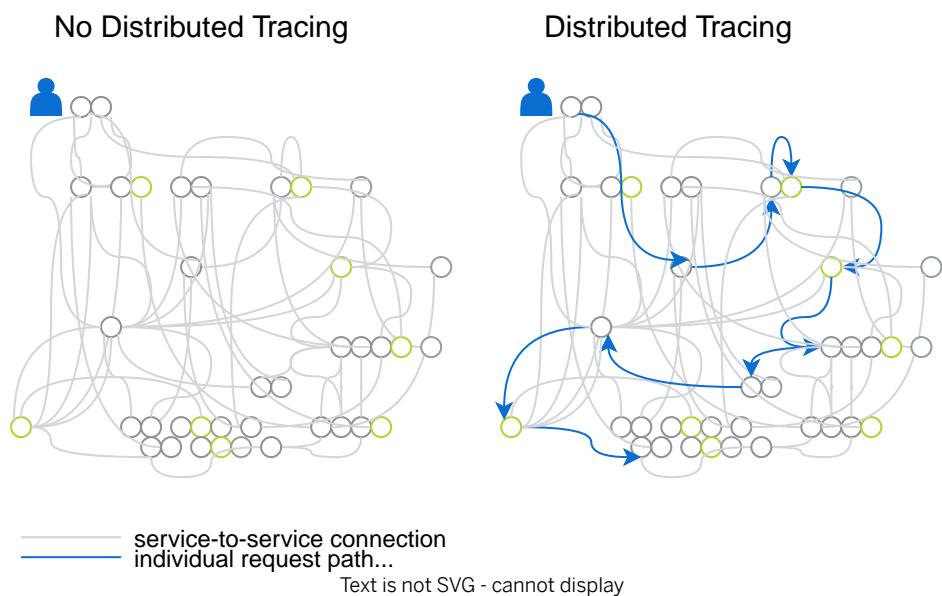
The Telemetry module supports you in collecting all relevant trace data in a Kyma cluster, enriches them and ships them to a backend for further analysis. Kyma modules like Istio or Serverless contribute traces transparently. You can choose among multiple [vendors for OTLP-based backends](#).

### Overview

Observability tools aim to show the big picture, no matter if you're monitoring just a few or many components. In a cloud-native microservice architecture, a user request often flows through dozens of different microservices. Logging and monitoring tools help to track the request's path. However, they treat each component or microservice in isolation. This individual treatment results in operational issues.

[Distributed tracing](#) charts out the transactions in cloud-native systems, helping you to understand the application behavior and relations between the frontend actions and backend implementation.

The following diagram shows how distributed tracing helps to track the request path:



The Telemetry module provides a trace gateway for the shipment of traces of any container running in the Kyma runtime.

You can configure the trace gateway with external systems using runtime configuration with a dedicated Kubernetes API ([CRD](#)) named `TracePipeline`.

The Trace feature is optional. If you don't want to use it, simply don't set up a `TracePipeline`.

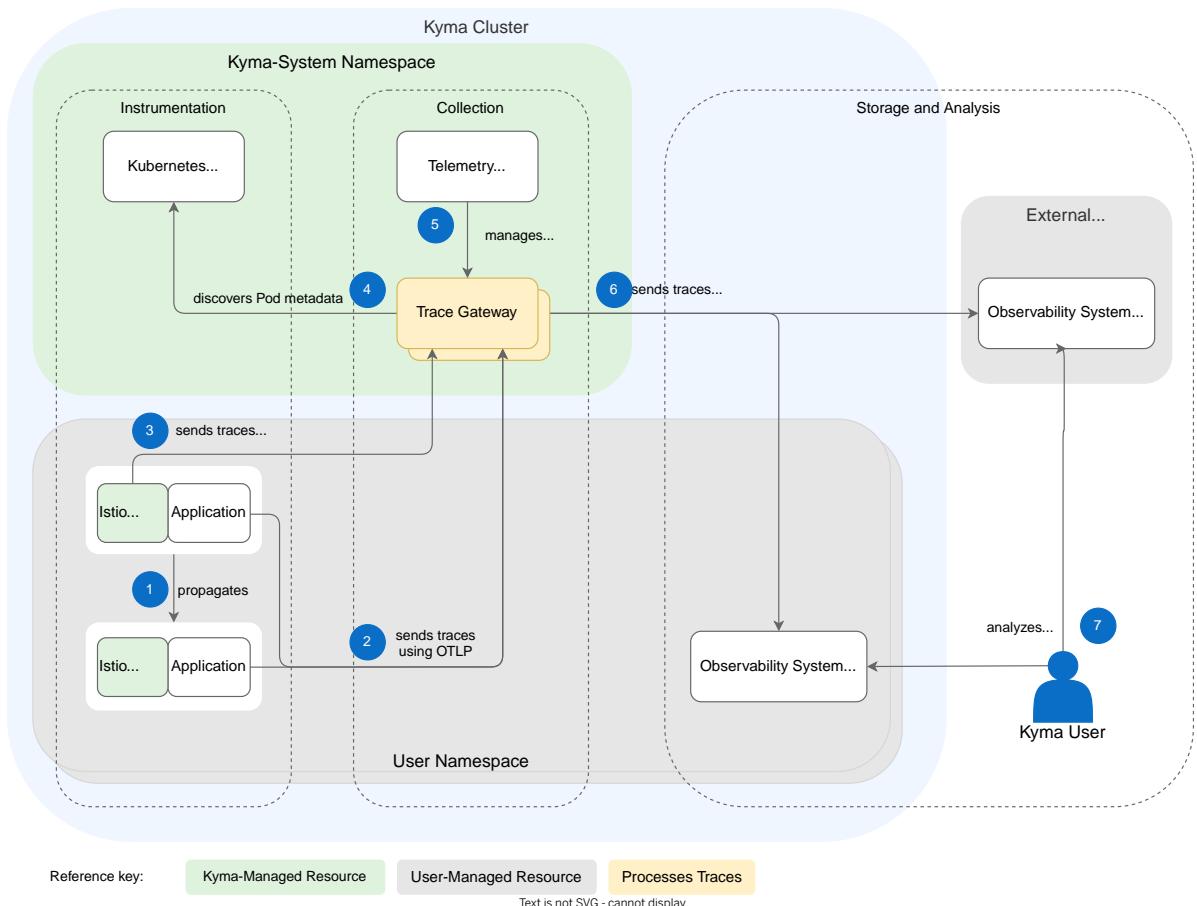
## Prerequisites

For the recording of a distributed trace, every involved component must propagate at least the trace context. For details, see [Trace Context](#).

- In Kyma, all modules involved in users' requests support the [W3C Trace Context](#) protocol. The involved Kyma modules are, for example, Istio, Serverless, and Eventing.
- Your application also must propagate the W3C Trace Context for any user-related activity. This can be achieved easily using the [Open Telemetry SDKs](#) available for all common programming languages. If your application follows that guidance and is part of the Istio Service Mesh, it's already outlined with dedicated span data in the trace data collected by the Kyma telemetry setup.
- Furthermore, your application must enrich a trace with additional span data and send these data to the cluster-central telemetry services. You can achieve this with [Open Telemetry SDKs](#).

## Architecture

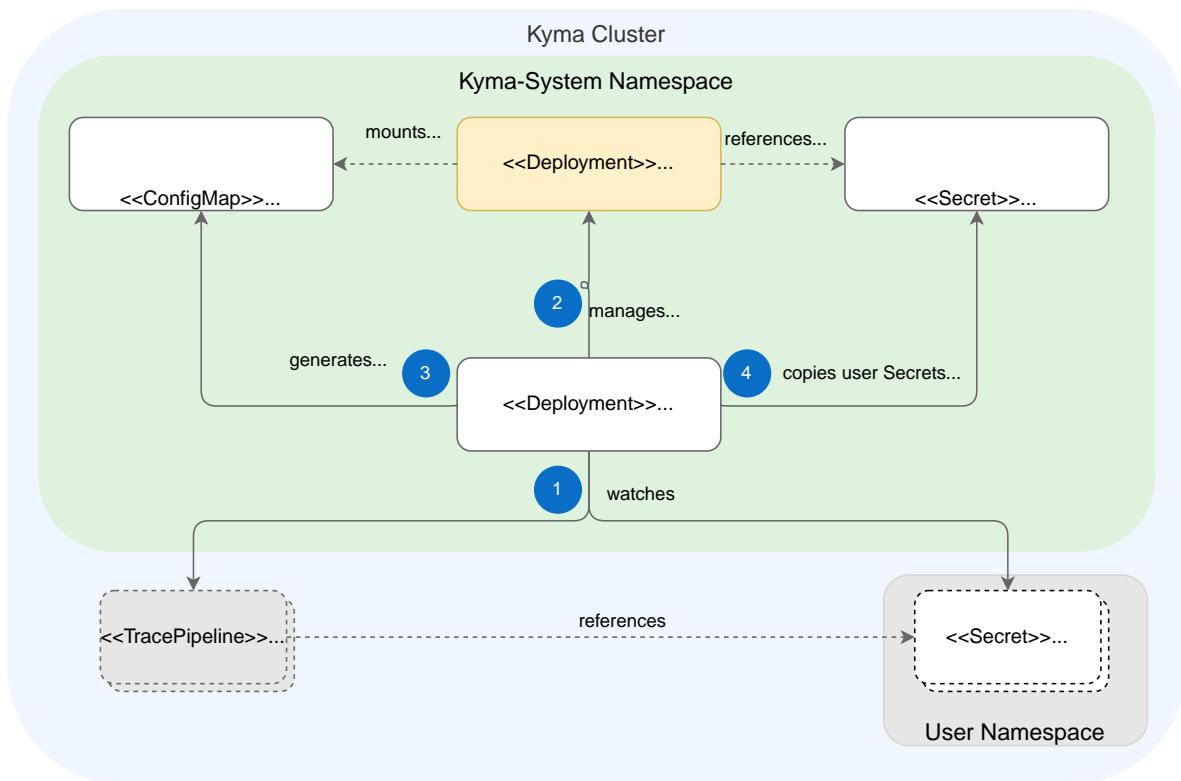
In the Kyma cluster, the Telemetry module provides a central deployment of an [OTel Collector](#) acting as a gateway. The gateway exposes endpoints to which all Kyma modules and users' applications should send the trace data.



1. An end-to-end request is triggered and populated across the distributed application. Every involved component propagates the trace context using the [W3C Trace Context](#) protocol.
2. After contributing a new span to the trace, the involved components send the related span data to the trace gateway using the [telemetry-otlp-traces](#) service. The communication happens based on the [OpenTelemetry Protocol \(OTLP\)](#) either using GRPC or HTTP.
3. Istio sends the related span data to the trace gateway as well.
4. The trace gateway discovers metadata that's typical for sources running on Kubernetes, like Pod identifiers, and then enriches the span data with that metadata.
5. Telemetry Manager configures the gateway according to the `TracePipeline` resource, including the target backend for the trace gateway. Also, it observes the trace flow to the backend and reports problems in the `TracePipeline` status.
6. The trace gateway sends the data to the observability system that's specified in your `TracePipeline` resource - either within the Kyma cluster, or, if authentication is set up, to an external observability backend.
7. You can analyze the trace data with your preferred backend system.

## Telemetry Manager

The `TracePipeline` resource is watched by Telemetry Manager, which is responsible for generating the custom parts of the OTel Collector configuration.



Reference key:

Kyma-Managed Resource

User-Managed Resource

Processes Traces

Text is not SVG - cannot display

1. Telemetry Manager watches all `<>TracePipeline>>...` resources and related Secrets.
2. Furthermore, Telemetry Manager takes care of the full lifecycle of the OTel Collector Deployment itself. Only if you defined a `<>TracePipeline>>...`, the collector is deployed.
3. Whenever the configuration changes, it validates the configuration and generates a new configuration for OTel Collector, where a `<>ConfigMap>>...` for the configuration is generated.
4. Referenced Secrets are copied into one Secret that is mounted to the OTel Collector as well.

## Trace Gateway

In a Kyma cluster, the trace gateway is the central component to which all components can send their individual spans. The gateway collects, enriches, and dispatches the data to the configured backend. For more information, see [Telemetry Gateways \[page 1927\]](#).

## Setting up a TracePipeline

In the following steps, you can see how to construct and deploy a typical `<>TracePipeline>>...`. Learn more about the available [parameters and attributes](#).

### 1. Create a TracePipeline

To ship traces to a new OTLP output, create a resource of the kind `TracePipeline` and save the file (named, for example, `tracepipeline.yaml`).

This configures the underlying OTel Collector with a pipeline for traces. It defines that the receiver of the pipeline is of the OTLP type and is accessible with the `telemetry-otlp-traces` service.

The default protocol is GRPC, but you can choose HTTP instead. Depending on the configured protocol, an `otlp` or an `otlphttp` exporter is used. Ensure that the correct port is configured as part of the endpoint. Typically, port [4317](#) is used for GRPC and port [4318](#) for HTTP.

- For GRPC, use:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

- For HTTP, use the `protocol` attribute:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      protocol: http
      endpoint:
        value: https://backend.example.com:4318
```

## 2. Enable Istio Tracing

By default, the tracing feature of the Istio module is disabled to avoid increased network utilization if there is no `TracePipeline`.

To activate the Istio tracing feature with a sampling rate of 5% (for recommendations, see [Istio](#)), use a resource similar to the following example:

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: tracing-default
  namespace: istio-system
spec:
  tracing:
    - providers:
      - name: "kyma-traces"
        randomSamplingPercentage: 5.00
```

## 3a. Add Authentication Details From Plain Text

To integrate with external systems, you must configure authentication details. You can use mutual TLS (mTLS), Basic Authentication, or custom headers:

- mTLS:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
```

```

output:
  otlp:
    endpoint:
      value: https://backend.example.com/otlp:4317
    tls:
      cert:
        value: |
          -----BEGIN CERTIFICATE-----
          ...
      key:
        value: |
          -----BEGIN RSA PRIVATE KEY-----
          ...

```

- Basic Authentication:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      authentication:
        basic:
          user:
            value: myUser
          password:
            value: myPwd

```

- Token-based authentication with custom headers:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      headers:
        - name: Authorization
          prefix: Bearer
          value: "myToken"

```

### 3b. Add Authentication Details From Secrets

Integrations into external systems usually need authentication details dealing with sensitive data. To handle that data properly in Secrets, TracePipeline supports the reference of Secrets.

Using the `valueFrom` attribute, you can map Secret keys for mutual TLS (mTLS), Basic Authentication, or with custom headers.

You can store the value of the token in the referenced Secret without any prefix or scheme, and you can configure it in the `headers` section of the TracePipeline. In the following example, the token has the prefix "Bearer".

- mTLS:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline

```

```

metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      tls:
        cert:
          valueFrom:
            secretKeyRef:
              name: backend
              namespace: default
              key: cert
        key:
          valueFrom:
            secretKeyRef:
              name: backend
              namespace: default
              key: key

```

- Basic Authentication:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        valueFrom:
          secretKeyRef:
            name: backend
            namespace: default
            key: endpoint
      authentication:
        basic:
          user:
            valueFrom:
              secretKeyRef:
                name: backend
                namespace: default
                key: user
          password:
            valueFrom:
              secretKeyRef:
                name: backend
                namespace: default
                key: password

```

- Token-based authentication with custom headers:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
      headers:
        - name: Authorization
          prefix: Bearer
          valueFrom:
            secretKeyRef:

```

```
name: backend
namespace: default
key: token
```

The related Secret must have the referenced name, be located in the referenced namespace, and contain the mapped key. See the following example:

```
kind: Secret
apiVersion: v1
metadata:
  name: backend
  namespace: default
stringData:
  endpoint: https://backend.example.com:4317
  user: myUser
  password: XXX
  token: YYY
```

## 4. Rotate the Secret

Telemetry Manager continuously watches the Secret referenced with the secretKeyRef construct. You can update the Secret's values, and Telemetry Manager detects the changes and applies the new Secret to the setup.

### → Tip

If you use a Secret owned by the [SAP BTP Service Operator](#), you can configure an automated rotation using a credentialsRotationPolicy with a specific rotationFrequency and don't have to intervene manually.

## 5. Deploy the Pipeline

To activate the TracePipeline, apply the `tracepipeline.yaml` resource file in your cluster:

```
kubectl apply -f tracepipeline.yaml
```

### Result

You activated a TracePipeline and traces start streaming to your backend.

To check that the pipeline is running, wait until the status conditions of the TracePipeline in your cluster have status True:

```
kubectl get tracepipeline
NAME      CONFIGURATION GENERATED   GATEWAY HEALTHY   FLOW HEALTHY
backend   True                      True            True
```

## Kyma Modules With Tracing Capabilities

Kyma bundles several modules that can be involved in user flows. Applications involved in a distributed trace must propagate the trace context to keep the trace complete. Optionally, they can enrich the trace with custom spans, which requires reporting them to the backend.

### Istio

The [Istio Module](#) [page 1868] is crucial in distributed tracing because it provides the [ingress gateway](#). Typically, this is where external requests enter the cluster scope and are enriched with trace context if it hasn't

happened earlier. Furthermore, every component that's part of the Istio Service Mesh runs an Istio proxy, which propagates the context properly but also creates span data. This is crucial in distributed tracing because it provides the

If Istio tracing is activated and taking care of trace propagation in your application, you get a complete picture of a trace, because every component automatically contributes span data. Also, Istio tracing is pre-configured to be based on the vendor-neutral [W3C Trace Context](#) protocol.

### ⚠ Caution

The provided Istio feature uses an API in alpha state, which may change in future releases.

The Istio module is configured with an [extension provider](#) called `kyma-traces`. To activate the provider on the global mesh level using the Istio [Telemetry API](#), place a resource to the `istio-system` namespace. The following code samples help setting up the Istio tracing feature:

- is crucial in distributedExtension provider: The following example configures all Istio proxies with the `kyma-traces` extension provider, which, by default, reports span data to the trace gateway of the Telemetry module.

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: tracing-default
  namespace: istio-system
spec:
  tracing:
    - providers:
      - name: "kyma-traces"
```

- Sampling rate: By default, the sampling rate is configured to 1%. That means that only 1 trace out of 100 traces is reported to the trace gateway, and all others are dropped. The sampling decision itself is propagated as part of the [trace context](#) so that either all involved components are reporting the span data of a trace, or none.

### → Tip

If you increase the sampling rate, you send more data to your tracing backend and cause much higher network utilization in the cluster. To reduce costs and performance impacts in a production setup, a very low percentage of around 5% is recommended.

To configure an “always-on” sampling, set the sampling rate to 100%:

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: tracing-default
  namespace: istio-system
spec:
  tracing:
    - providers:
      - name: "kyma-traces"
        randomSamplingPercentage: 100.00
```

- Namespaces or workloads: If you need specific settings for individual namespaces or workloads, place additional Telemetry resources. If you don't want to report spans at all for a specific workload, activate the `disableSpanReporting` flag with the selector expression.

```
apiVersion: telemetry.istio.io/v1
```

```

kind: Telemetry
metadata:
  name: tracing-default
  namespace: my-namespace
spec:
  selector:
    matchLabels:
      kubernetes.io/name: "my-app"
  tracing:
    - providers:
      - name: "kyma-traces"
        randomSamplingPercentage: 100.00

```

- Trace context without spans: To enable the propagation of the [W3C Trace Context](#) only, without reporting any spans (so the actual tracing feature is disabled), you must enable the `kyma-traces` provider with a sampling rate of 0. With this configuration, you get the relevant trace context into the access logs without any active trace reporting.

```

apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: tracing-default
  namespace: istio-system
spec:
  tracing:
    - providers:
      - name: "kyma-traces"
        randomSamplingPercentage: 0

```

## Eventing

The [Eventing](#) module uses the [CloudEvents](#) protocol (which natively supports the [W3C Trace Context](#)). Because of that, it propagates trace context properly. However, it doesn't enrich a trace with more advanced span data.

## Serverless

By default, all engines for the [Serverless](#) module integrate the [Open Telemetry SDK](#). Thus, the used middlewares are configured to automatically propagate the trace context for chained calls.

Because the Telemetry endpoints are configured by default, Serverless also reports custom spans for incoming and outgoing requests. You can [customize Function traces](#) to add more spans as part of your Serverless source code.

## Operations

A [TracePipeline](#) runs several OTel Collector instances in your cluster. This Deployment serves OTLP endpoints and ships received data to the configured backend.

The Telemetry module ensures that the OTel Collector instances are operational and healthy at any time, for example, with buffering and retries. However, there may be situations when the instances drop traces, or cannot handle the trace load.

To detect and fix such situations, check the pipeline status and check out [Troubleshooting \[page 1948\]](#).

## Limitations

- **Throughput:** Assuming an average span with 40 attributes with 64 characters, the maximum throughput is 4200 span/sec  $\approx$  15.000.000 spans/hour. If this limit is exceeded, spans are refused. To increase the maximum throughput, manually scale out the gateway by increasing the number of replicas.
- **Unavailability of Output:** For up to 5 minutes, a retry for data is attempted when the destination is unavailable. After that, data is dropped.
- **No Guaranteed Delivery:** The used buffers are volatile. If the OTel Collector instance crashes, trace data can be lost.
- **Multiple TracePipeline Support:** The maximum amount of TracePipeline resources is 3.
- **System Span Filtering:** System-related spans reported by Istio are filtered out without the opt-out option, for example:
  - Any communication of applications to the Telemetry gateways
  - Any communication from the gateways to backends

## Troubleshooting

### No Spans Arrive at the Backend

**Symptom:** In the TracePipeline status, the TelemetryFlowHealthy condition has status AllDataDropped.

**Cause:** Incorrect backend endpoint configuration (such as using the wrong authentication credentials), or the backend is unreachable.

**Solution:**

1. Check the telemetry-trace-gateway Pods for error logs by calling `kubectl logs -n kyma-system {POD_NAME}`.
2. Check if the backend is up and reachable.
3. Fix the errors.

### Not All Spans Arrive at the Backend

**Symptom:**

- The backend is reachable and the connection is properly configured, but some spans are refused.
- In the TracePipeline status, the TelemetryFlowHealthy condition has status SomeDataDropped.

**Cause:** It can happen due to a variety of reasons - for example, the backend is limiting the ingestion rate.

**Solution:**

1. Check the telemetry-trace-gateway Pods for error logs by calling `kubectl logs -n kyma-system {POD_NAME}`. Also, check your observability backend to investigate potential causes.
2. If the backend is limiting the rate by refusing spans, try the options described in [Gateway Buffer Filling Up](#).
3. Otherwise, take the actions appropriate to the cause indicated in the logs.

### Custom Spans Don't Arrive at the Backend, but Istio Spans Do

**Cause:** Your SDK version is incompatible with the OTel Collector version.

**Solution:**

1. Check which SDK version you are using for instrumentation.
2. Investigate whether it is compatible with the OTel Collector version.
3. If required, upgrade to a supported SDK version.

## Trace Backend Shows Fewer Traces than Expected

**Cause:** By default, only 1% of the requests are sent to the trace backend for trace recording.

**Solution:**

To see more traces in the trace backend, increase the percentage of requests by changing the default settings.

If you just want to see traces for one particular request, you can manually force sampling:

1. Create a `values.yaml` file.

The following example sets the value to 60, which means 60% of the requests are sent to the tracing backend.

```
apiVersion: telemetry.istio.io/v1
kind: Telemetry
metadata:
  name: kyma-traces
  namespace: istio-system
spec:
  tracing:
    - providers:
      - name: "kyma-traces"
randomSamplingPercentage: 60
```

2. To override the default percentage, change the value for the `randomSamplingPercentage` attribute.
3. Deploy the `values.yaml` to your existing Kyma installation.

## Gateway Buffer Filling Up

**Symptom:** In the `TracePipeline` status, the `TelemetryFlowHealthy` condition has status `BufferFillingUp`.

**Cause:** The backend export rate is too low compared to the gateway ingestion rate.

**Solution:**

- Option 1: Increase the maximum backend ingestion rate - for example, by scaling out the SAP Cloud Logging instances.
- Option 2: Reduce the emitted spans in your applications.

## Gateway Throttling

**Symptom:**

- In the `TracePipeline` status, the `TelemetryFlowHealthy` condition has status `GatewayThrottling`.
- Also, your application might have error logs indicating a refusal for sending traces to the gateway.

**Cause:** Gateway cannot receive spans at the given rate.

**Solution:** Manually scale out the gateway by increasing the number of replicas for the trace gateway. See [Module Configuration and Status \[page 1926\]](#).

## 4.3.1.5.5 Metrics

The goal of the Telemetry module is to support you in collecting all relevant metrics of a workload in a Kyma cluster and ship them to a backend for further analysis. Kyma modules like [Istio Module \[page 1868\]](#) or [Serverless](#) contribute metrics instantly, and the Telemetry module enriches the data. You can choose among multiple [vendors for OTLP-based backends](#).

### Overview

Observability is all about exposing the internals of the components belonging to a distributed application and making that data analysable at a central place. While application logs and traces usually provide request-oriented data, metrics are aggregated statistics exposed by a component to reflect the internal state. Typical statistics like the amount of processed requests, or the amount of registered users, can be very useful to monitor the current state and also the health of a component. Also, you can define proactive and reactive alerts if metrics are about to reach thresholds, or if they already passed thresholds.

The Telemetry module provides a metric gateway and, optionally, an agent for the collection and shipment of metrics of any container running in the Kyma runtime.

You can configure the metric gateway with external systems using runtime configuration with a dedicated Kubernetes API ([CRD](#)) named `MetricPipeline`. The Metric feature is optional. If you don't want to use it, simply don't set up a `MetricPipeline`.

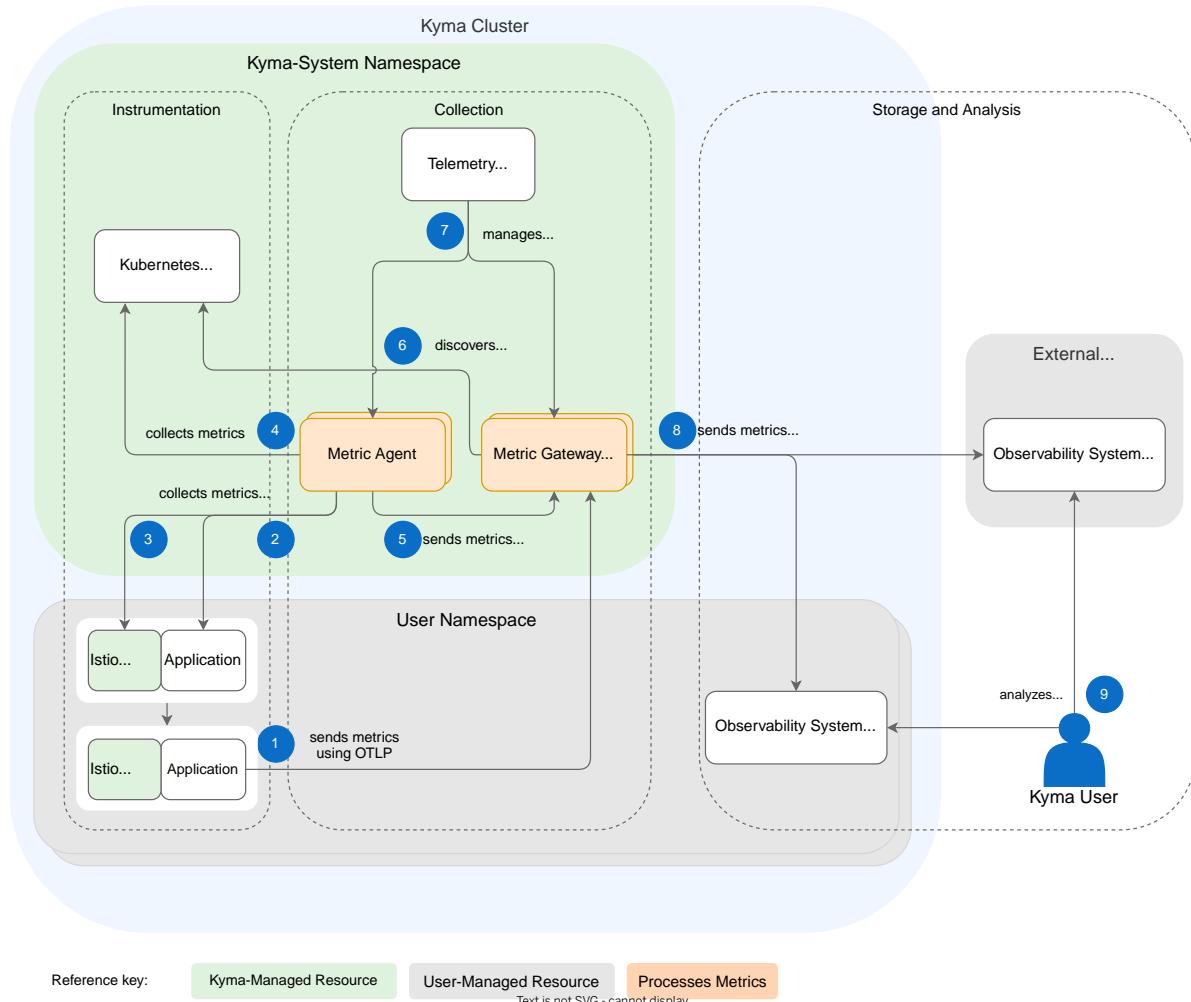
### Prerequisites

- Before you can collect metrics data from a component, it must expose (or instrument) the metrics. Typically, it instruments specific metrics for the used language runtime (like Node.js) and custom metrics specific to the business logic. Also, the exposure can be in different formats, like the pull-based Prometheus format or the [push-based OTLP format](#).
- If you want to use Prometheus-based metrics, you must have instrumented your application using a library like the [Prometheus client library](#), with a port in your workload exposed serving as a Prometheus metrics endpoint.
- For the instrumentation, you typically use an SDK, namely the [Prometheus client libraries](#) or the [Open Telemetry SDKs](#). Both libraries provide extensions to activate language-specific auto-instrumentation like for Node.js, and an API to implement custom instrumentation.

### Architecture

In the Telemetry module, a central in-cluster Deployment of an [OTel Collector](#) acts as a gateway. The gateway exposes endpoints for the [OpenTelemetry Protocol \(OTLP\)](#) for GRPC and HTTP-based communication using the dedicated `telemetry-otlp-metrics` service, to which all Kyma modules and users' applications send the metrics data.

Optionally, the Telemetry module provides a DaemonSet of an OTel Collector acting as an agent. This agent can pull metrics of a workload and the Istio sidecar in the [Prometheus pull-based format](#) and can provide runtime-specific metrics for the workload.

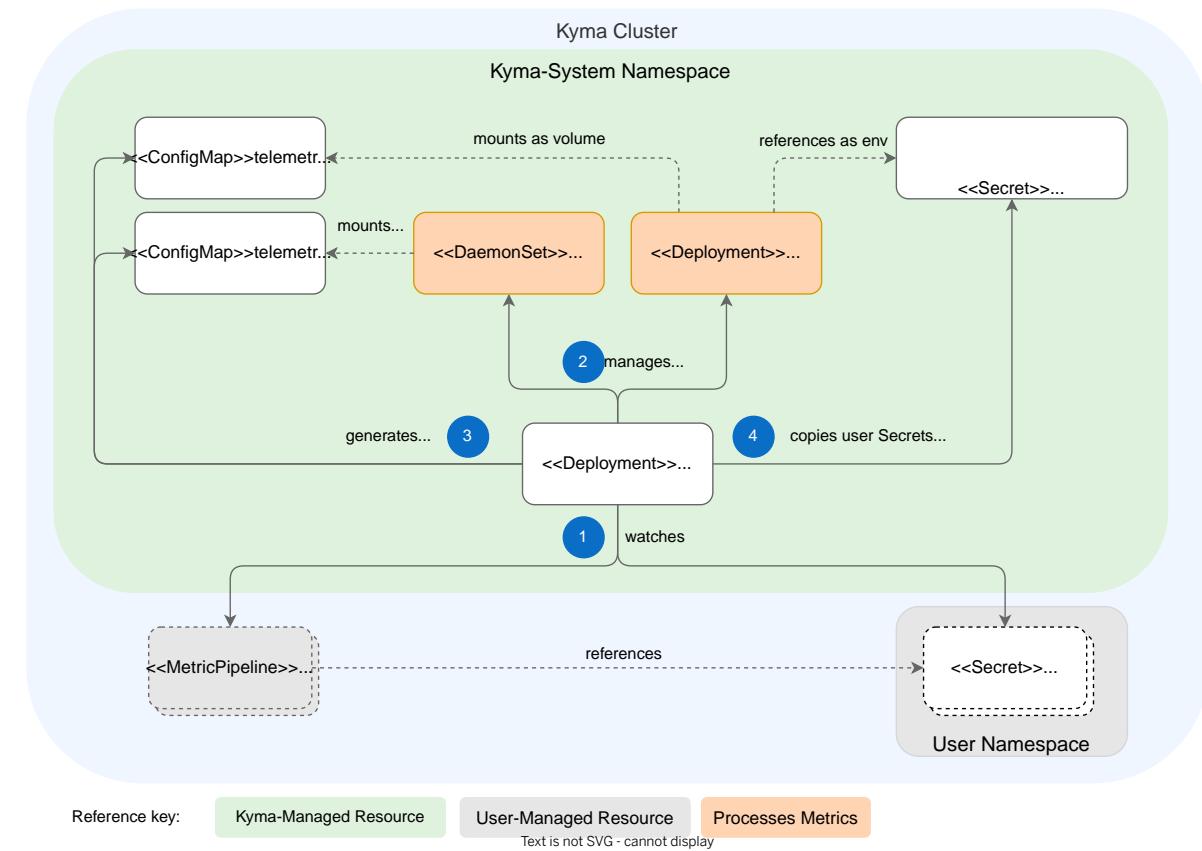


1. An application (exposing metrics in OTLP) sends metrics to the central metric gateway service.
2. An application (exposing metrics in Prometheus protocol) activates the agent to scrape the metrics with an annotation-based configuration.
3. Additionally, you can activate the agent to pull metrics of each Istio sidecar.
4. The agent supports collecting metrics from the Kubelet and Kubernetes API Server.
5. The agent converts and pushes all collected metric data to the gateway in OTLP.
6. The gateway discovers the metadata and enriches all received data with typical metadata of the source by communicating with the Kubernetes API Server. Furthermore, it filters data according to the pipeline configuration.
7. Telemetry Manager configures the agent and gateway according to the `MetricPipeline` resource specification, including the target backend for the metric gateway. Also, it observes the metrics flow to the backend and reports problems in the `MetricPipeline` status.
8. The metric gateway sends the data to the observability system that's specified in your `MetricPipeline` resource - either within the Kyma cluster, or, if authentication is set up, to an external observability backend.

- You can analyze the metric data with your preferred backend system.

## Telemetry Manager

The MetricPipeline resource is watched by Telemetry Manager, which is responsible for generating the custom parts of the OTel Collector configuration.



- Telemetry Manager watches all MetricPipeline resources and related Secrets.
- Furthermore, Telemetry Manager takes care of the full lifecycle of the gateway Deployment and the agent DaemonSet itself. Only if you defined a MetricPipeline, the gateway and agent are deployed.
- Whenever the configuration changes, Telemetry Manager validates it and generates a single configuration for the gateway and agent.
- Referenced Secrets are copied into one Secret that is mounted to the gateway as well.

## Metric Gateway

In a Kyma cluster, the metric gateway is the central component to which all components can send their individual metrics. The gateway collects, enriches, and dispatches the data to the configured backend. For more information, see [Telemetry Gateways \[page 1927\]](#).

## Metric Agent

If a MetricPipeline configures a feature in the `input` section, an additional DaemonSet is deployed acting as an agent. The agent is also based on an [OTel Collector](#) and encompasses the collection and conversion of Prometheus-based metrics. Hereby, the workload puts a `prometheus.io/scrape` annotation on the specification of the Pod or service, and the agent collects it. The agent sends all data in OTLP to the central gateway.

## Setting up a MetricPipeline

In the following steps, you can see how to construct and deploy a typical MetricPipeline. Learn more about the available [parameters and attributes](#).

### 1. Create a MetricPipeline

To ship metrics to a new OTLP output, create a resource of the kind MetricPipeline and save the file (named, for example, metricpipeline.yaml).

This configures the underlying Otel Collector of the gateway with a pipeline for metrics. It defines that the receiver of the pipeline is of the OTLP type and is accessible with the telemetry-otlp-metrics service.

The default protocol is GRPC, but you can choose HTTP instead. Depending on the configured protocol, an otlp or an otlphttp exporter is used. Ensure that the correct port is configured as part of the endpoint. Typically, port **4317** is used for GRPC and port **4318** for HTTP.

- For GRPC, use:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

- For HTTP, use the protocol attribute:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      protocol: http
      endpoint:
        value: https://backend.example.com:4318
```

### 2a. Add Authentication Details From Plain Text

To integrate with external systems, you must configure authentication details. You can use mutual TLS (mTLS), Basic Authentication, or custom headers:

- mTLS:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      tls:
        cert:
          value: |
-----BEGIN CERTIFICATE-----
```

```
    ...
key:
  value: |
-----BEGIN RSA PRIVATE KEY-----
  ...
```

- Basic Authentication:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      authentication:
        basic:
          user:
            value: myUser
          password:
            value: myPwd
```

- Token-based authentication with custom headers:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      headers:
        - name: Authorization
          prefix: Bearer
          value: "myToken"
```

## 2b. Add Authentication Details From Secrets

Integrations into external systems usually need authentication details dealing with sensitive data. To handle that data properly in Secrets, MetricsPipeline supports the reference of Secrets.

Using the `valueFrom` attribute, you can map Secret keys for mutual TLS (mTLS), Basic Authentication, or with custom headers.

You can store the value of the token in the referenced Secret without any prefix or scheme, and you can configure it in the `headers` section of the `MetricPipeline`. In this example, the token has the prefix “Bearer”.

- mTLS:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com/otlp:4317
      tls:
        cert:
          valueFrom:
```

```

    secretKeyRef:
      name: backend
      namespace: default
      key: cert
  key:
    valueFrom:
      secretKeyRef:
        name: backend
        namespace: default
        key: key

```

- Basic Authentication:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        valueFrom:
          secretKeyRef:
            name: backend
            namespace: default
            key: endpoint
  authentication:
    basic:
      user:
        valueFrom:
          secretKeyRef:
            name: backend
            namespace: default
            key: user
      password:
        valueFrom:
          secretKeyRef:
            name: backend
            namespace: default
            key: password

```

- Token-based authentication with custom headers:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
  headers:
    - name: Authorization
      prefix: Bearer
      valueFrom:
        secretKeyRef:
          name: backend
          namespace: default
          key: token

```

The related Secret must have the referenced name, be located in the referenced namespace, and contain the mapped key. See the following example:

```

kind: Secret
apiVersion: v1

```

```

metadata:
  name: backend
  namespace: default
stringData:
  endpoint: https://backend.example.com:4317
  user: myUser
  password: XXX
  token: YYY

```

### 3. Rotate the Secret

Telemetry Manager continuously watches the Secret referenced with the secretKeyRef construct. You can update the Secret's values, and Telemetry Manager detects the changes and applies the new Secret to the setup.

→ Tip

If you use a Secret owned by the [SAP BTP Service Operator](#), you can configure an automated rotation using a credentialsRotationPolicy with a specific rotationFrequency and don't have to intervene manually.

### 4. Activate Prometheus-Based Metrics

→ Remember

For the following approach, you must have instrumented your application using a library like the [Prometheus client library](#), with a port in your workload exposed serving as a Prometheus metrics endpoint.

To enable collection of Prometheus-based metrics, define a MetricPipeline that has the `prometheus` section enabled as input:

```

apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    prometheus:
      enabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317

```

The Metric agent is configured with a generic scrape configuration, which uses annotations to specify the endpoints to scrape in the cluster.

For metrics ingestion to start automatically, simply apply the following annotations either to a Service that resolves your metrics port, or directly to the Pod:

Prometheus Metrics Annotations

Annotation Key	Example Values	Default Value	Description
<code>prometheus.io/scrape</code> (mandatory)	true, false	none	Controls whether Prometheus automatically scrapes metrics from this target.

Annotation Key	Example Values	Default Value	Description
prometheus.io/port (mandatory)	8080, 9100	none	Specifies the port where the metrics are exposed.
prometheus.io/path	/metrics, /custom_metrics	/metrics	Defines the HTTP path where Prometheus can find metrics data.
prometheus.io/ scheme	http, https	If Istio is active, https is supported; otherwise, only http is available.  The default scheme is http unless an Istio sidecar is present, denoted by the label <b>security.istio.io/tlsMode=istio</b> , in which case https becomes the default.	Determines the protocol used for scraping metrics — either HTTPS with mTLS or plain HTTP.

If you're running the Pod targeted by a Service with Istio, Istio must be able to derive the [appProtocol](#) from the Service port definition; otherwise the communication for scraping the metric endpoint cannot be established. You must either prefix the port name with the protocol like in `http-metrics`, or explicitly define the `appProtocol` attribute. For example, see the following Service configuration:

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    prometheus.io/port: "8080"
    prometheus.io/scrape: "true"
    name: sample
spec:
  ports:
  - name: http-metrics
    appProtocol: http
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: sample
  type: ClusterIP
```

## ⓘ Note

The Metric agent can scrape endpoints even if the workload is a part of the Istio service mesh and accepts mTLS communication. However, there's a constraint: For scraping through HTTPS, Istio must configure the workload using "STRICT" mTLS mode. Without "STRICT" mTLS mode, you can set up scraping through HTTP by applying the annotation `prometheus.io/scheme=http`. For related troubleshooting, see [Log Entry: Failed to Scrape Prometheus Endpoint](#).

## 5. Monitor Pipeline Health

By default, a `MetricPipeline` emits metrics about the health of all pipelines managed by the Telemetry module. Based on these metrics, you can track the status of every individual pipeline and set up alerting for it.

## Metrics for Pipelines and the Telemetry Module

Metric	Description	Availability
kyma.resource.status.conditions	Value represents status of different conditions reported by the resource. Possible values are 1 ("True"), 0 ("False"), and -1 (other status values)	Available for both, the pipelines and the Telemetry resource
kyma.resource.status.state	Value represents the state of the resource (if present)	Available for the Telemetry resource

## Metric Attributes for Monitoring

Attribute	Description
metric.attributes.Type	Type of the condition
metric.attributes.status	Status of the condition
metric.attributes.reason	Contains a programmatic identifier indicating the reason for the condition's last transition

To set up alerting, use an alert rule. In the following example, the alert is triggered if metrics are not delivered to the backend:

```
min by (k8s_resource_name)
((kyma_resource_status_conditions{type="TelemetryFlowHealthy",k8s_resource_kind="metricpipelines"})) == 0
```

## 6. Activate Runtime Metrics

To enable collection of runtime metrics, define a `MetricPipeline` that has the `runtime` section enabled as input:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    runtime:
      enabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

By default, metrics for all resources (Pod, container, Node, Volume, DaemonSet, Deployment, StatefulSet, and Job) are collected.

To enable or disable the collection of metrics for a specific resource, use the `resources` section in the `runtime` input.

The following example collects only DaemonSet, Deployment, StatefulSet, and Job metrics:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    runtime:
      resources:
        daemonset: true
        deployment: true
        statefulset: true
        job: true
```

```

enabled: true
resources:
  pod:
    enabled: false
  container:
    enabled: false
  node:
    enabled: false
  volume:
    enabled: false
  daemonset:
    enabled: true
  deployment:
    enabled: true
  statefulset:
    enabled: true
  job:
    enabled: true
output:
  otlp:
    endpoint:
      value: https://backend.example.com:4317

```

#### Collected Metrics per Resource

Resource	From the <b>kubletstatsreceiver</b> ↗	From the <b>k8sclusterreceiver</b> ↗
<b>Pod</b>	<ul style="list-style-type: none"> <li>• k8s.pod.cpu.capacity</li> <li>• k8s.pod.cpu.usage</li> <li>• k8s.pod.filesystem.available</li> <li>• k8s.pod.filesystem.capacity</li> <li>• k8s.pod.filesystem.usage</li> <li>• k8s.pod.memory.available</li> <li>• k8s.pod.memory.major_page_faults</li> <li>• k8s.pod.memory.page_faults</li> <li>• k8s.pod.memory.rss</li> <li>• k8s.pod.memory.usage</li> <li>• k8s.pod.memory.working_set</li> <li>• k8s.pod.network.errors</li> <li>• k8s.pod.network.io</li> </ul>	<ul style="list-style-type: none"> <li>• k8s.pod.phase</li> </ul>

Resource	From the <a href="#">kubletstatsreceiver</a> 	From the <a href="#">k8sclusterreceiver</a> 
<b>Container</b>	<ul style="list-style-type: none"> <li>• container.cpu.time</li> <li>• container.cpu.usage</li> <li>• container.filesystem.available</li> <li>• container.filesystem.capacity</li> <li>• container.filesystem.usage</li> <li>• container.memory.available</li> <li>• container.memory.major_page_faults</li> <li>• container.memory.page_faults</li> <li>• container.memory.rss</li> <li>• container.memory.usage</li> <li>• container.memory.working_set</li> </ul>	<ul style="list-style-type: none"> <li>• k8s.container.cpu_request</li> <li>• k8s.container.cpu_limit</li> <li>• k8s.container.memory_request</li> <li>• k8s.container.memory_limit</li> </ul>
<b>Node</b>	<ul style="list-style-type: none"> <li>• k8s.node.cpu.usage</li> <li>• k8s.node.filesystem.available</li> <li>• k8s.node.filesystem.capacity</li> <li>• k8s.node.filesystem.usage</li> <li>• k8s.node.memory.available</li> <li>• k8s.node.memory.usage</li> <li>• k8s.node.memory.rss</li> <li>• k8s.node.memory.working_set</li> </ul>	-
<b>Volume</b>	<ul style="list-style-type: none"> <li>• k8s.volume.available</li> <li>• k8s.volume.capacity</li> <li>• k8s.volume.inodes</li> <li>• k8s.volume.inodes.free</li> <li>• k8s.volume.inodes.used</li> </ul>	-
<b>Deployment</b>	-	<ul style="list-style-type: none"> <li>• k8s.deployment.available</li> <li>• k8s.deployment.desired</li> </ul>
<b>DaemonSet</b>	-	<ul style="list-style-type: none"> <li>• k8s.daemonset.current_scheduled_nodes</li> <li>• k8s.daemonset.desired_scheduled_nodes</li> <li>• k8s.daemonset.misscheduled_nodes</li> <li>• k8s.daemonset.ready_nodes</li> </ul>
<b>StatefulSet</b>	-	<ul style="list-style-type: none"> <li>• k8s.statefulset.current_pods</li> <li>• k8s.statefulset.desired_pods</li> <li>• k8s.statefulset.ready_pods</li> <li>• k8s.statefulset.updated_pods</li> </ul>

Resource	From the <a href="#">kubletstatsreceiver</a>	From the <a href="#">k8sclusterreceiver</a>
Job	-	<ul style="list-style-type: none"> <li>k8s.job.active_pods</li> <li>k8s.job.desired_successful_pods</li> <li>k8s.job.failed_pods</li> <li>k8s.job.max_parallel_pods</li> <li>k8s.job.successful_pods</li> </ul>

## 7. Activate Istio Metrics

To enable collection of Istio metrics, define a `MetricPipeline` that has the `istio` section enabled as input:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    istio:
      enabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

With this, the agent starts collecting all Istio metrics from Istio sidecars.

## 8. Deactivate OTLP Metrics

By default, `otlp` input is enabled.

To drop the push-based OTLP metrics that are received by the Metric gateway, define a `MetricPipeline` that has the `otlp` section disabled as an input:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    istio:
      enabled: true
    otlp:
      disabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

With this, the agent starts collecting all Istio metrics from Istio sidecars, and the push-based OTLP metrics are dropped.

## 9. Add Filters

To filter metrics by namespaces, define a `MetricPipeline` that has the `namespaces` section defined in one of the inputs. For example, you can specify the namespaces from which metrics are collected or the namespaces from which metrics are dropped. Learn more about the available [parameters and attributes](#).

- The following example collects runtime metrics **only** from the `foo` and `bar` namespaces:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    runtime:
      enabled: true
      namespaces:
        include:
          - foo
          - bar
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

- The following example collects runtime metrics from all namespaces **except** the `foo` and `bar` namespaces:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    runtime:
      enabled: true
      namespaces:
        exclude:
          - foo
          - bar
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

## ⓘ Note

The default settings depend on the input:

If no namespace selector is defined for the `prometheus` or `runtime` input, then metrics from system namespaces are **excluded** by default.

However, if the namespace selector is not defined for the `istio` and `otlp` input, then metrics from system namespaces are **included** by default.

## 10. Activate Diagnostic Metrics

If you use the `prometheus` or `istio` input, for every metric source typical scrape metrics are produced, such as `up`, `scrape_duration_seconds`, `scrape_samples_scraped`, `scrape_samples_post_metric_relabeling`, and `scrape_series_added`.

By default, they are disabled.

If you want to use them for debugging and diagnostic purposes, you can activate them. To activate diagnostic metrics, define a `MetricPipeline` that has the `diagnosticMetrics` section defined.

- The following example collects diagnostic metrics **only** for input `istio`:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
```

```
metadata:
  name: backend
spec:
  input:
    istio:
      enabled: true
      diagnosticMetrics:
        enabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

- The following example collects diagnostic metrics **only** for input `prometheus`:

```
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: backend
spec:
  input:
    prometheus:
      enabled: true
      diagnosticMetrics:
        enabled: true
  output:
    otlp:
      endpoint:
        value: https://backend.example.com:4317
```

### ⓘ Note

Diagnostic metrics are only available for inputs `prometheus` and `istio`. Learn more about the available [parameters and attributes](#).

## 11. Deploy the Pipeline

To activate the `MetricPipeline`, apply the `metricpipeline.yaml` resource file in your cluster:

```
kubectl apply -f metricpipeline.yaml
```

### Result

You activated a `MetricPipeline` and metrics start streaming to your backend.

To check that the pipeline is running, wait until the status conditions of the `MetricPipeline` in your cluster have status `True`:

### ↔ Output Code

```
kubectl get metricpipeline
NAME      CONFIGURATION GENERATED   GATEWAY HEALTHY   AGENT HEALTHY   FLOW
HEALTHY
backend   True                      True            True
```

## Operations

A `MetricPipeline` runs several OTel Collector instances in your cluster. This Deployment serves OTLP endpoints and ships received data to the configured backend.

The Telemetry module ensures that the OTel Collector instances are operational and healthy at any time, for example, with buffering and retries. However, there may be situations when the instances drop metrics, or cannot handle the metric load.

To detect and fix such situations, check the pipeline status and check out [Troubleshooting \[page 1964\]](#).

## Limitations

- **Throughput:** Assuming an average metric with 20 metric data points and 10 labels, the default metric `gateway` setup has a maximum throughput of 34K metric data points/sec. If more data is sent to the gateway, it is refused. To increase the maximum throughput, manually scale out the gateway by increasing the number of replicas for the Metric gateway.  
The metric `agent` setup has a maximum throughput of 14K metric data points/sec per instance. If more data must be ingested, it is refused. If a metric data endpoint emits more than 50.000 metric data points per scrape loop, the metric agent refuses all the data.
- **Load Balancing With Istio:** To ensure availability, the metric gateway runs with multiple instances. If you want to increase the maximum throughput, use manual scaling and enter a higher number of instances. By design, the connections to the gateway are long-living connections (because OTLP is based on gRPC and HTTP/2). For optimal scaling of the gateway, the clients or applications must balance the connections across the available instances, which is automatically achieved if you use an Istio sidecar. If your application has no Istio sidecar, the data is always sent to one instance of the gateway.
- **Unavailability of Output:** For up to 5 minutes, a retry for data is attempted when the destination is unavailable. After that, data is dropped.
- **No Guaranteed Delivery:** The used buffers are volatile. If the gateway or agent instances crash, metric data can be lost.
- **Multiple MetricPipeline Support:** The maximum amount of `MetricPipeline` resources is 3.

## Troubleshooting

### No Metrics Arrive at the Backend

#### Symptom:

- No metrics arrive at the backend.
- In the `MetricPipeline` status, the `TelemetryFlowHealthy` condition has status `AllDataDropped`.

**Cause:** Incorrect backend endpoint configuration (such as using the wrong authentication credentials) or the backend is unreachable.

#### Solution:

1. Check the `telemetry-metric-gateway` Pods for error logs by calling `kubectl logs -n kyma-system {POD_NAME}`.

2. Check if the backend is up and reachable.
3. Fix the errors.

## Not All Metrics Arrive at the Backend

### Symptom:

- The backend is reachable and the connection is properly configured, but some metrics are refused.
- In the MetricPipeline status, the TelemetryFlowHealthy condition has status SomeDataDropped.

**Cause:** It can happen due to a variety of reasons - for example, the backend is limiting the ingestion rate.

### Solution:

1. Check the telemetry-metric-gateway Pods for error logs by calling kubectl logs -n kyma-system {POD\_NAME}. Also, check your observability backend to investigate potential causes.
2. If backend is limiting the rate by refusing metrics, try the options described in [Gateway Buffer Filling Up](#).
3. Otherwise, take the actions appropriate to the cause indicated in the logs.

## Only Istio Metrics Arrive at the Backend

**Symptom:** Custom metrics don't arrive at the backend, but Istio metrics do.

**Cause:** Your SDK version is incompatible with the OTEL Collector version.

### Solution:

1. Check which SDK version you are using for instrumentation.
2. Investigate whether it is compatible with the OTEL Collector version.
3. If required, upgrade to a supported SDK version.

## Log Entry: Failed to Scrape Prometheus Endpoint

**Symptom:** Custom metrics don't arrive at the destination. The OTEL Collector produces log entries saying "Failed to scrape Prometheus endpoint", such as the following example:

### ↔ Output Code

```
2023-08-29T09:53:07.123Z warn internal/transaction.go:111 Failed to scrape
Prometheus endpoint {"kind": "receiver", "name": "prometheus/app-pods",
"data_type": "metrics", "scrape_timestamp": 1693302787120, "target_labels":
"__name__=\\"up\\", instance=\\"10.42.0.18:8080\\", job=\\"app-pods\\\"}"}
```

**Cause 1:** The workload is not configured to use "STRICT" mTLS mode. For details, see [Activate Prometheus-Based Metrics](#).

**Solution 1:** You can either set up "STRICT" mTLS mode or HTTP scraping:

- Configure the workload using "STRICT" mTLS mode (for example, by applying a corresponding PeerAuthentication).
- Set up scraping through HTTP by applying the prometheus.io/scheme=http annotation.

**Cause 2:** The Service definition enabling the scrape with Prometheus annotations does not reveal the application protocol to use in the port definition. For details, see [Activate Prometheus-Based Metrics](#).

**Solution 2:** Define the application protocol in the Service port definition by either prefixing the port name with the protocol, like in http-metrics or define the appProtocol attribute.

**Cause 3:** A deny-all NetworkPolicy was created in the workload namespace, which prevents that the agent can scrape metrics from annotated workloads.

**Solution 3:** Create a separate NetworkPolicy to explicitly let the agent scrape your workload using the `telemetry.kyma-project.io/metric-scrape` label. For example, see the following NetworkPolicy configuration:

```
yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-traffic-from-agent
spec:
  podSelector:
    matchLabels:
      app.kubernetes.io/name: "annotated-workload" # <your workload here>
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          kubernetes.io/metadata.name: kyma-system
    podSelector:
      matchLabels:
        telemetry.kyma-project.io/metric-scrape: "true"
  policyTypes:
  - Ingress
```

## Gateway Buffer Filling Up

**Symptom:** In the MetricPipeline status, the TelemetryFlowHealthy condition has status BufferFillingUp.

**Cause:** The backend export rate is too low compared to the gateway ingestion rate.

**Solution:**

- Option 1: Increase maximum backend ingestion rate. For example, by scaling out the SAP Cloud Logging instances.
- Option 2: Reduce emitted metrics by re-configuring the MetricPipeline (for example, by disabling certain inputs or applying namespace filters).
- Option 3: Reduce emitted metrics in your applications.

## Gateway Throttling

**Symptom:** In the MetricPipeline status, the TelemetryFlowHealthy condition has status GatewayThrottling.

**Cause:** Gateway cannot receive metrics at the given rate.

**Solution:** Manually scale out the gateway by increasing the number of replicas for the Metric gateway. See [Module Configuration and Status \[page 1926\]](#).

## 4.3.1.5.6 Integrate with SAP Cloud Logging

Learn how to configure the Telemetry module to ingest application and access logs as well as distributed trace data and metrics in instances of SAP Cloud Logging.

### Prerequisites

- Kyma as the target deployment environment.
- The Telemetry module is added. For details, see [Add and Delete a Kyma Module \[page 3012\]](#).
- If you want to use Istio access logs, make sure that the Istio module is added.
- An instance of [SAP Cloud Logging](#) with OpenTelemetry enabled to ingest distributed traces.

#### → Tip

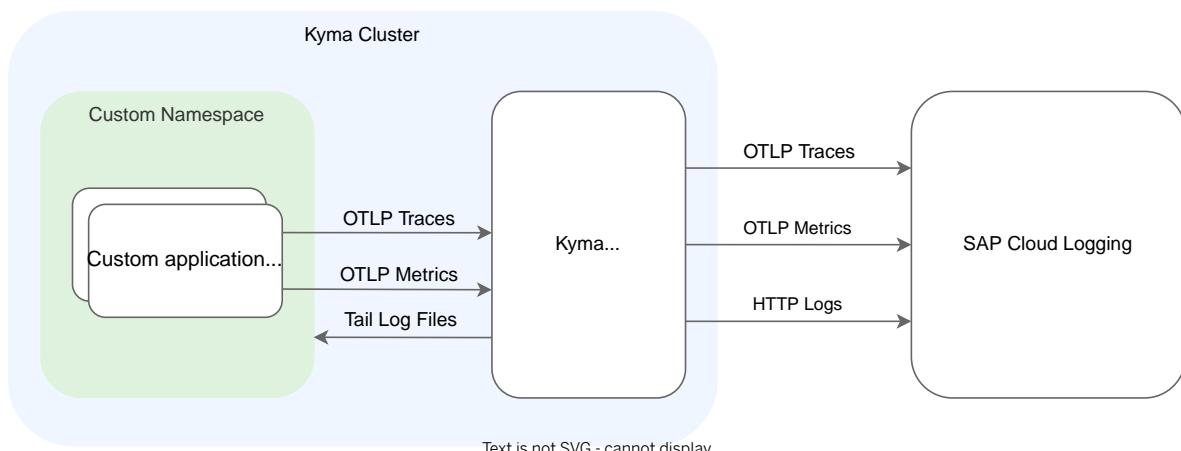
Create the instance with the SAP BTP service operator (see [Create an SAP Cloud Logging Instance through SAP BTP Service Operator](#)), because it takes care of creation and rotation of the required Secret. However, you can choose any other method of creating the instance and the Secret, as long as the parameter for OTLP ingestion is enabled in the instance. For details, see [Configuration Parameters](#).

- A Secret in the respective namespace in the Kyma cluster, holding the credentials and endpoints for the instance. In the following example, the Secret is named “sap-cloud-logging” and the namespace “sap-cloud-logging-integration”, as illustrated in the [secret-example.yaml](#).
- Kubernetes CLI (kubectl) (see [Install the Kubernetes Command Line Tool](#)).
- UNIX shell or Windows Subsystem for Linux (WSL) to execute commands.

### Context

The Telemetry module supports shipping logs and ingesting distributed traces as well as metrics from applications and the Istio service mesh to SAP Cloud Logging. Furthermore, you can set up Kyma dashboard integration and use SAP Cloud Logging alerts and dashboards.

SAP Cloud Logging is an instance-based and environment-agnostic observability service to store, visualize, and analyze logs, metrics, and traces.



# Ship Logs to SAP Cloud Logging

You can set up shipment of applications and access logs to SAP Cloud Logging. The following instructions distinguish application logs and access logs, which can be configured independently.

## Procedure

### Set Up Application Logs

1. Deploy the LogPipeline for application logs with the following script:

```
kubectl apply -n sap-cloud-logging-integration -f - <<EOF
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: LogPipeline
metadata:
  name: sap-cloud-logging-application-logs
spec:
  input:
    application:
      containers:
        exclude:
          - istio-proxy
  output:
    http:
      dedot: true
      host:
        valueFrom:
          secretKeyRef:
            name: sap-cloud-logging
            namespace: sap-cloud-logging-integration
            key: ingest-mtls-endpoint
  tls:
    cert:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-mtls-cert
    key:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-mtls-key
  uri: /customindex/kyma
EOF
```

2. Wait for the LogPipeline to be in the *Running* state. To check the state, run: `kubectl get logpipelines`.

### Set Up Access Logs

By default, Istio sidecar injection and Istio access logs are disabled in Kyma. To analyze access logs of your workload in the default SAP Cloud Logging dashboards shipped for SAP BTP, Kyma runtime, you must enable them:

3. Enable Istio sidecar injection for your workload. See [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#).
4. Enable Istio access logs for your workload. See [Configuring Istio Access Logs \[page 1883\]](#).

5. Deploy the LogPipeline for Istio access logs and enable access logs in Kyma with the following script:

```
kubectl apply -n sap-cloud-logging-integration -f - <<EOF
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: LogPipeline
metadata:
  name: sap-cloud-logging-access-logs
spec:
  input:
    application:
      containers:
        include:
          - istio-proxy
  output:
    http:
      dedot: true
      host:
        valueFrom:
          secretKeyRef:
            name: sap-cloud-logging
            namespace: sap-cloud-logging-integration
            key: ingest-mtls-endpoint
  tls:
    cert:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-mtls-cert
    key:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-mtls-key
  uri: /customindex/istio-envoy-kyma
EOF
```

6. Wait for the LogPipeline to be in the *Running* state. To check the state, run: `kubectl get logpipelines`.

## Ship Distributed Traces to SAP Cloud Logging

You can set up ingestion of distributed traces from applications and the Istio service mesh to the OTLP endpoint of the SAP Cloud Logging service instance.

### Procedure

1. Deploy the Istio Telemetry resource with the following script:

```
kubectl apply -n istio-system -f - <<EOF
apiVersion: telemetry.istio.io/v1alpha1
kind: Telemetry
metadata:
  name: tracing-default
spec:
  tracing:
```

```

    - providers:
      - name: "kyma-traces"
        randomSamplingPercentage: 1.0
EOF

```

The default configuration has the `randomSamplingPercentage` property set to 1.0, meaning it samples 1% of all requests. To change the sampling rate, adjust the property to the desired value, up to 100 percent.

2. Deploy the `TracePipeline` with the following script:

```

kubectl apply -n sap-cloud-logging-integration -f - <<EOF
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: TracePipeline
metadata:
  name: sap-cloud-logging
spec:
  output:
    otlp:
      endpoint:
        valueFrom:
          secretKeyRef:
            name: sap-cloud-logging
            namespace: sap-cloud-logging-integration
            key: ingest-otlp-endpoint
    tls:
      cert:
        valueFrom:
          secretKeyRef:
            name: sap-cloud-logging
            namespace: sap-cloud-logging-integration
            key: ingest-otlp-cert
      key:
        valueFrom:
          secretKeyRef:
            name: sap-cloud-logging
            namespace: sap-cloud-logging-integration
            key: ingest-otlp-key
EOF

```

3. Wait for the `TracePipeline` to be in the *Running* state. To check the state, run: `kubectl get tracepipelines`.

## Ship Metrics to SAP Cloud Logging

You can set up ingestion of metrics from applications and the Istio service mesh to the OTLP endpoint of the SAP Cloud Logging service instance.

### Procedure

1. Deploy the `MetricPipeline` with the following script:

```

kubectl apply -n sap-cloud-logging-integration -f - <<EOF
apiVersion: telemetry.kyma-project.io/v1alpha1
kind: MetricPipeline
metadata:
  name: sap-cloud-logging
spec:

```

```

input:
  prometheus:
    enabled: false
  istio:
    enabled: false
  runtime:
    enabled: false
output:
  otlp:
    endpoint:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-otlp-endpoint
  tls:
    cert:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-otlp-cert
    key:
      valueFrom:
        secretKeyRef:
          name: sap-cloud-logging
          namespace: sap-cloud-logging-integration
          key: ingest-otlp-key
EOF

```

By default, the `MetricPipeline` assures that a gateway is running in the cluster to push OTLP metrics.

2. If you want to use additional metric collection, configure the presets under `input`.

For the available options, see [Metrics \[page 1950\]](#).

3. Wait for the `MetricPipeline` to be in the *Running* state. To check the state, run: `kubectl get metricpipelines`.

## Set Up Kyma Dashboard Integration

For easier access from the Kyma dashboard, add links to the navigation under [SAP Cloud Logging](#), and add deep links to the [Pod](#), [Deployment](#), and [Namespace](#) views.

### Procedure

1. Apply the ConfigMap:

```

kubectl apply
-f https://raw.githubusercontent.com/kyma-project/telemetry-manager/main/docs/
user/integration/sap-cloud-logging/kyma-dashboard-configmap.yaml

```

2. If your Secret has a different name or namespace, then download the file first and adjust the namespace and name accordingly in the `dataSources` section of the file.

# Use SAP Cloud Logging Alerts

Learn how to define and import recommended alerts for SAP Cloud Logging. The following alerts are based on JSON documents defining a `Monitor` for the alerting plugin.

## Procedure

1. Define a destination, which will be used by all your alerts.
2. To import a monitor, use the development tools of the SAP Cloud Logging dashboard.
3. Execute `POST _plugins/_alerting/monitors`, followed by the contents of the respective JSON file.
4. Depending on the pipelines you are using, enable some or all of the following alerts:

Recommended Alerts

Category	File	Description
SAP Cloud Logging	<a href="#">OpenSearch cluster health</a>	The OpenSearch cluster might become unhealthy, which is indicated by a red status using the <a href="#">cluster health API</a> .
Kyma Telemetry Integration	<a href="#">Application log ingestion</a>	The LogPipeline for shipping application logs might lose connectivity to SAP Cloud Logging, with the effect that no application logs are ingested anymore.
Kyma Telemetry Integration	<a href="#">Access log ingestion</a>	The LogPipeline for shipping access logs might lose connectivity to SAP Cloud Logging, with the effect that no access logs are ingested anymore.
Kyma Telemetry Integration	<a href="#">Trace ingestion</a>	The TracePipeline for shipping traces might lose connectivity to SAP Cloud Logging, with the effect that no traces are ingested anymore.
Kyma Telemetry Integration	<a href="#">Metric ingestion</a>	The MetricPipeline for shipping metrics might lose connectivity to SAP Cloud Logging, with the effect that no metrics are ingested anymore.
Kyma Telemetry Integration	<a href="#">Kyma Telemetry Status</a>	The Telemetry module might report a non-ready state indicating a configuration or data flow problem.

5. Edit notification action: Add the destination and adjust the intervals and thresholds to your needs.
6. Verify that the new monitor definition is listed among the SAP Cloud Logging alerts.

# Use SAP Cloud Logging Dashboards

You can view logs, traces, and metrics in SAP Cloud Logging dashboards:

## Procedure

- To view the traffic and application logs, use the SAP Cloud Logging dashboards prefixed with `Kyma_`, which are based on both kinds of log ingestion: application and access logs.
- To view distributed traces, use the OpenSearch plugin [Observability](#).
- To view the container- and Pod-related metrics collected by the `MetricPipeline runtime` input, use the dashboard [\[OTel\] K8s Container Metrics](#).
- To view the Kubernetes Node-related metrics collected by the `MetricPipeline runtime` input, manually import the file [K8s Nodes](#).
- To view the Kubernetes Volume-related metrics collected by the `MetricPipeline runtime` input, manually import the file [K8s Volumes](#).
- To view the Kubernetes Workload-related metrics collected by the `MetricPipeline runtime` input, manually import the file [K8s Workloads](#).
- To view the status of the SAP Cloud Logging integration with the Kyma Telemetry module, manually import the file [Kyma Telemetry Status](#).
- To use the dashboard for Istio metrics of Pods that have an active Istio sidecar injection (collected by the `MetricPipeline istio` input), manually import the file [Kyma Istio Service Metrics](#).

## 4.3.2 Access a Kyma Instance Using kubectl

As an alternative to managing Kyma with a graphical user interface, Kyma dashboard, you can also use the Kubernetes command-line tool, kubectl.

## Prerequisites

You have a Kyma instance created in your subaccount of the SAP BTP cockpit.

## Context

To start using kubectl, follow these steps:

## Procedure

1. Install [kubectl oidc-login](#). Follow the original instruction from the project repository. Use the relevant commands for macOS, Linux, or Windows users.

### → Recommendation

On Windows, we recommend using the release binaries to install the kubectl oidc-login plugin. The installation of the kubectl oidc-login plugin using Chocolatey and Krew could cause issues.

2. In the SAP BTP cockpit, in your subaccount [Overview](#), go to the [Kyma Environment](#) section, and click on the [KubeconfigURL](#) link to download `kubeconfig.yaml`.
3. In the terminal, export the kubeconfig file.
  - On macOS, run:

```
export KUBECONFIG={KUBECONFIG_FILE_PATH}
```
  - On Windows (PowerShell), run:

```
$ENV:KUBECONFIG=" {KUBECONFIG_FILE_PATH}"
```

## Results

Now you can manage your Kyma instance using kubectl.

### → Tip

If you authenticate in the Kyma environment with an Identity Provider using OpenID Connect, the downloaded `kubeconfig.yaml` can be used indefinitely to re-authenticate.

## Related Information

[Configure a Custom Identity Provider for Kyma \[page 3134\]](#)

## 4.3.3 Using SAP BTP Services in the Kyma Environment

With the Kyma environment you can connect SAP BTP services to your cluster and manage them using the SAP BTP Operator module.

### Related Information

- [SAP BTP Operator Module \[page 1894\]](#)
- [Preconfigured Credentials and Access \[page 1897\]](#)
- [Creating Service Instances and Service Bindings \[page 1899\]](#)
- [Working with Multiple Subaccounts \[page 1902\]](#)
- [Instance-Level Mapping \[page 1903\]](#)
- [Namespace-Level Mapping \[page 1906\]](#)
- [Passing Parameters \[page 1907\]](#)
- [Rotating Service Bindings \[page 1909\]](#)
- [Formatting Service Binding Secrets \[page 1910\]](#)
- [Deleting Service Bindings and Service Instances \[page 1915\]](#)

### 4.3.3.1 Creating and Managing Services Using Kyma dashboard

Create and manage Service Instances and Service Bindings using Kyma dashboard.

#### Context

Use Kyma dashboard to create and manage your resources.

#### Procedure

1. In the [Namespace](#) view, go to [Service Management](#)→[Service Instances](#).  
2. Create Service Instance using the required service details.  
You see the status PROVISIONED.
3. Go to [Service Management](#)→[Service Bindings](#) and create Service Binding, choosing your instance name from the dropdown list.  
You see the status PROVISIONED.

## Results

You can now use a given service in your Kyma cluster.

### 4.3.3.2 Creating and Managing Services Using kubectl

Create and manage Service Instances and Service Bindings using kubectl.

#### Context

Use kubectl to create and manage your resources.

#### Procedure

1. Create a Service Instance:

```
kubectl create -f - <<EOF
apiVersion: services.cloud.sap.com/v1
kind: ServiceInstance
metadata:
  name: {INSTANCE_NAME}
  namespace: {NAMESPACE}
spec:
  serviceOfferingName: {NAME_FROM_SERVICE_MARKETPLACE}
  servicePlanName: {PLAN_FROM_SERVICE_MARKETPLACE}
  externalName: {INSTANCE_NAME}
EOF
```

2. To see the output, run:

```
kubectl get serviceinstances.services.cloud.sap.com {INSTANCE_NAME} -o yaml
```

You can see the status `created` and the message `ServiceInstance provisioned successfully.`

3. Create a Service Binding:

```
kubectl create -f - <<EOF
apiVersion: services.cloud.sap.com/v1
kind: ServiceBinding
metadata:
  name: {BINDING_NAME}
  namespace: {NAMESPACE}
spec:
  serviceInstanceId: {INSTANCE_NAME}
  externalName: {BINDING_NAME}
  secretName: {BINDING_NAME}
  parameters:
    key1: val1
    key2: val2
EOF
```

4. To see the output, run:

```
kubectl get servicebindings.services.cloud.sap.com {BINDING_NAME} -o yaml
```

You can see the status created and the message ServiceBinding provisioned successfully.

## Results

You can now use a given service in your Kyma cluster. To see credentials, run:

```
kubectl get secret {BINDING_NAME} -o yaml
```

## Next Steps

To clean up your resources, run:

```
kubectl delete servicebindings.services.cloud.sap.com {BINDING_NAME}  
kubectl delete serviceinstances.services.cloud.sap.com {INSTANCE_NAME}
```

### 4.3.4 Deploy Workloads in the Kyma Environment to Extend SAP Systems

Access the Kyma environment and start creating extensions for SAP systems.

## Prerequisites

- You have added the Application Connector and Serverless modules. To learn how to do it, see [Add and Delete a Kyma Module \[page 3012\]](#).
- If you want to expose and secure your workload, the default Istio and API Gateway modules must be added.
- If you want your workload to react to events, the Eventing module must be added.
- If you want to use [CloudEvents](#), make sure that your ecosystem supports it.

## Context

You can extend a given SAP system in the Kyma environment with workloads. There are two kinds of workloads

- Functions and microservices:

- Functions are simple code snippets that are called when the Function is triggered. They implement the exact business logic you define in the code, and you don't need to worry about containerization and deployment in Kyma runtime. Because Kyma Serverless is taking care of that, using Functions is convenient and fast.
- You can also deploy your extension code as a microservice. In this variant, you must contain your microservice code in a Docker image and deploy it in Kyma runtime. You need more Kubernetes experience but get more control over the shape of your extension application, because you can separate the tasks into smaller pieces that interact with each other as loosely coupled, independently deployable units of code.

Whichever workload variant you choose, you can make it part of the SAP system extension. You can do this in two ways. Depending on your solution's needs, choose one or both of them:

- Expose your workload's API using API Rule and make it accessible to other components of your ecosystem with REST calls.
- Subscribe your workload to events emitted from the SAP systems and extend them.

### ⓘ Note

Most of the linked tutorials contain steps that you can perform both on the Kyma dashboard and in the terminal after downloading the kubeconfig file with the cluster configuration.

Follow these steps to get familiar with workloads and learn how to use them.

## Procedure

1. Connect the SAP system you want to extend ([Extending SAP Solutions Using Automated Configurations](#)).
2. Choose a namespace in your cluster and create a simple Function ([Create an Inline Function](#)) or a microservice ([Create a Workload](#)).
3. If you want to expose your workload outside the cluster, read [Expose a Workload](#).

### ⓘ Note

For more details on the available security options in Kyma, see [Configure Authorization \(OAuth2, JWT\)](#).

4. If you want your workload to react to events, read [Trigger the Workload with an Event](#).

To subscribe to events with Kyma, you must create a [Subscription](#) including the following parameters:

- **Application name:** The name of the external Application connected to Kyma runtime. Typically, it starts with `mp-*`. This name can be found in the UI under [Integration](#) [Applications](#). It must be bound to the namespace.
- **Event name:** The event name depends on your CX solution:
  - **SAP Commerce Cloud** - for custom events, check the configuration of your [Destination Target](#) for the Kyma runtime

- SAP Field Service Management
- SAP Sales Cloud / SAP Service Cloud - you can check the event specification in [Event Notification Monitoring](#).
- **Event version:** v1 for legacy events or v2 for CloudEvents

#### ⚠ Caution

Your workload must be error-free, otherwise event loss could occur. If you're facing problems, check the Kyma [Troubleshooting documentation](#).

## Related Information

[Kyma: Serverless](#)

### 4.3.5 Expose and Secure a Workload with a JSON Web Token

Learn how to expose a workload and secure it with a JSON Web Token (JWT). To get the token, create a client credentials application using SAP Cloud Identity Services - Identity Authentication.

## Prerequisites

- You have an SAP Cloud Identity Services - Identity Authentication tenant.
- You have created an OpenID Connect Application in your Identity Authentication tenant. See [Create OpenID Connect Application for JWT Bearer Flow](#).
- You have configured a Secret for API Authentication and saved the values of Client ID and Client Secret. See [Configure Secrets for API Authentication](#).
- You have the default Istio and API Gateway modules in your cluster.
- You have a workload deployed.

## Context

To interact with a workload that is secured using an API Rule of the JWT type, you need an access token. Follow these steps to obtain a JWT using an Identity Authentication tenant. Then, use either kubectl or Kyma dashboard to expose and secure your workload.

### 4.3.5.1 Get a JSON Web Token

Follow the instructions to obtain a JWT.

#### Procedure

1. Export the name of your Identity Authentication instance and your client credentials as environment variables. Run:

```
export  
IDENTITY_AUTHENTICATION_INSTANCE={YOUR_IDENTITY_AUTHENTICATION_INSTANCE_NAME}  
export CLIENT_ID={YOUR_CLIENT_ID}  
export CLIENT_SECRET={YOUR_CLIENT_SECRET}
```

2. Encode your client credentials and export them as an environment variable:

```
export ENCODED_CREDENTIALS=$(echo -n "$CLIENT_ID:$CLIENT_SECRET" | base64)
```

3. Get values of the **token\_endpoint** and **jwks\_uri** parameters:

```
curl  
-s https://$IDENTITY_AUTHENTICATION_INSTANCE.accounts400.ondemand.com/.well-known/openid-configuration
```

##### ⓘ Note

You need the value of the **jwks\_uri** parameter to secure your workload using an APIRule of the JWT type.

4. Export the value of the **token\_endpoint** as an environment variable:

```
export TOKEN_ENDPOINT={YOUR_TOKEN_ENDPOINT}
```

5. Get the JWT:

```
curl -X POST "$TOKEN_ENDPOINT" -d "grant_type=client_credentials" -d  
"client_id=$CLIENT_ID" -H "Content-Type: application/x-www-form-urlencoded"  
-H "Authorization: Basic $ENCODED_CREDENTIALS"
```

6. Export the JWT as an environment variable:

```
export ACCESS_TOKEN={YOUR_ACCESS_TOKEN}
```

#### Results

You got a JWT that you can use to interact with a workload.

## 4.3.5.2 Expose and Secure a Workload with a JWT Using Kyma Dashboard

Use Kyma dashboard to expose a workload and secure it with a JWT.

### Procedure

1. Create a namespace.
2. Go to [Discovery and Network](#) → [API Rules](#) and select [Create](#).
3. Provide the following configuration details:

Parameter	Value
Name	httpbin
Service Name	httpbin
Port	8000
Host	httpbin.{DOMAIN_TO_EXPOSE_WORKLOADS}
	Replace {DOMAIN_TO_EXPOSE_WORKLOADS} with the name of your domain.
Access Strategy Handler	jwt

4. To access the secured service, call it using the JWT.

```
curl -ik https://httpbin.$DOMAIN_TO_EXPOSE_WORKLOADS/headers -H  
"Authorization: Bearer $ACCESS_TOKEN"
```

This call returns the 200 OK response code.

### Results

You have secured your service using an APIRule with the JWT access strategy. You can use the JWT to interact with the service.

### 4.3.5.3 Expose and Secure a Workload with a JWT Using kubectl

Use kubectl to expose a workload and secure it with a JWT.

#### Procedure

1. Create a namespace and export its value as an environment variable:

```
export NAMESPACE={NAMESPACE_NAME}
kubectl create ns $NAMESPACE
```

2. Export the following values as environment variables:

```
export DOMAIN_TO_EXPOSE_WORKLOADS={DOMAIN_NAME}
export JWKS_URI={YOUR_JWKS_URI}
```

3. To expose and secure your workload, create the APIRule in your namespace:

```
cat <<EOF | kubectl apply -f -
apiVersion: gateway.kyma-project.io/v1beta1
kind: APIRule
metadata:
  name: httpbin
  namespace: $NAMESPACE
spec:
  service:
    name: httpbin
    port: 8000
    host: httpbin.$DOMAIN_TO_EXPOSE_WORKLOADS
  gateway: kyma-system/kyma-gateway
  rules:
    - accessStrategies:
        - handler: jwt
          config:
            jwks_urls:
              - $JWKS_URI
      methods:
        - GET
      path: /.*
EOF
```

4. To access the secured service, call it using the JWT:

```
curl -ik https://httpbin.$DOMAIN_TO_EXPOSE_WORKLOADS/headers -H
"Authorization: Bearer $ACCESS_TOKEN"
```

This call returns the 200 OK response code.

#### Results

You have secured your service using an APIRule with the JWT access strategy. You can use the JWT to interact with the service.

## 4.3.6 Develop Resilient Applications in the Kyma Runtime

All SAP BTP, Kyma runtime production plans ensure high availability. To benefit from the high-availability setup, make sure the architecture and deployment of your application comply with resiliency best practices. Use the following guidelines for Kubernetes and microservice-based applications to develop a stable application. Read further to find some Istio-related resiliency tips.

### Resilient Applications Deployed on Kubernetes

To develop resilient applications deployed on Kubernetes, use:

- Kubernetes built-in health checks. For more information, read how to [Configure Liveness, Readiness and Startup Probes](#).
- Deployment across multiple availability zones for high availability. For more details, see the [blog post](#).
- The [Keda Module \[page 1918\]](#) or [Kubernetes autoscaling](#) to automatically adjust the number of replicas based on resource usage.
- [Rolling updates](#) to minimize downtime during application updates. If you don't provide any configuration, rolling updates are by default enabled in your SAP BTP, Kyma runtime.
- BTP backing services, such as [PostgreSQL](#) or [Object Store](#) on SAP BTP. If you have to store some application data on SAP BTP, Kyma runtime, use persistent storage to ensure data is not lost in the event of a pod failure.
- [Resource quotas](#) to prevent resource starvation and ensure fair resource allocation.
- The [Telemetry Module \[page 1921\]](#) to configure a logging and monitoring solution to identify and troubleshoot issues in real time.

### Resilient Microservice-Based Applications

The [Istio Module \[page 1868\]](#) and Istio traffic management features help you achieve inbuilt microservices resiliency. Moreover, to develop resilient microservice-based applications, use:

- Istio's circuit breaker and retry features to prevent service overloads and recover from service failures.
- Distributed tracing with [Istio's observability features](#) to monitor the flow of requests across microservices and identify performance and error issues by shipping logs to SAP Cloud Logging (for details, see [Integrate with SAP Cloud Logging \[page 1967\]](#)).
- The [Eventing module](#) for asynchronous communication patterns, such as messaging queues, to decouple services and reduce dependencies. See how to [Configure SAP Event Mesh for Kyma Eventing \[page 1986\]](#).
- Health check endpoints as a part of your microservice. For more details, read [Kubernetes API health endpoints](#).
- A centralized logging and monitoring solution to identify issues across multiple microservices and quickly troubleshoot them.
- A clear and well-defined boundary in your microservice's design to minimize the impact of failures and enable easy replacement or scaling of individual services.

- Istio's service mesh to implement mutual TLS and authorization policies to secure communication between microservices. For more details, read the [blog post](#).
- Network resiliency using Istio's [traffic management](#) features to test how your microservices respond to various failure scenarios, such as slow or failing services.
- The [Telemetry Module \[page 1921\]](#) to ship your Istio's access logs and metrics to your logging backend to monitor traffic patterns and quickly troubleshoot issues.

## Related Information

[Improve resiliency of your applications deployed on Kyma runtime by using multiple availability zones](#)

[Developing enterprise-grade applications with mutual TLS easy with SAP BTP, Kyma runtime and BTP destinations](#)

[SAP BTP Kyma API Rules with destinations and a managed approuter](#)

[Protect your Kyma workloads from eavesdropping with JWT access strategy and BTP destinations](#)

[Protect your Kyma workloads from eavesdropping with introspection and BTP destinations](#)

### 4.3.7 Choose a Backend for Kyma Eventing

The event service in Kyma runtime supports two backends: NATS and SAP Event Mesh. Learn how to set up your preferred eventing backend.

#### Prerequisites

- You have added the Kyma Eventing module. See [Add and Delete a Kyma Module \[page 3012\]](#).
- If you want to use NATS, you have added the Kyma NATS module.
- If you want to use SAP Event Mesh, you have configured it for Kyma Eventing. See [Configure SAP Event Mesh for Kyma Eventing \[page 1986\]](#).

#### Context

You can send and receive events in Kyma with the built-in eventing infrastructure. By default, Kyma clusters have no eventing backend defined. You must set up an eventing backend, either based on the NATS technology or SAP Event Mesh.

## Procedure

1. Log in to Kyma dashboard. The URL is in the [Overview](#) section of your subaccount.
2. Go to the `kyma-system` namespace.
3. Go to  [Kyma](#)  [Eventing](#), open the `eventing` resource, and choose [Edit](#).
4. Under **Backend Type**, select your preferred backend.
5. If you selected [EventMesh](#), go to [Event Mesh Secret](#), and select the namespace and name of your SAP Event Mesh service binding.

### → Remember

If you selected [NATS](#), make sure the Kyma NATS module is enabled.

6. Choose [Save](#).

## Results

You have set up your eventing backend for Kyma.

If you want to choose another backend, edit the `eventing` resource again, select the preferred backend, and save your changes.

You can no longer return to an empty eventing backend.

## Next Steps

Learn more about the Kyma [Eventing Module](#)  and follow the [Eventing Tutorials](#) .

## Related Information

[Configure SAP Event Mesh for Kyma Eventing \[page 1986\]](#)

[SAP Event Mesh](#)

## 4.3.7.1 Configure SAP Event Mesh for Kyma Eventing

If you want to use SAP Event Mesh as backend for Kyma Eventing, you must first set up the credentials.

### Prerequisites

- The Kyma Eventing module is added to your cluster. See [Add and Delete a Kyma Module \[page 3012\]](#).
- The Kyma SAP BTP Operator module is added to your cluster.
- You are using an enterprise account.

### Procedure

1. Log in to Kyma dashboard. The URL is in the [Overview](#) section of your subaccount.
2. To generate an SAP Event Mesh Secret, select a namespace and go to [Service Management](#) [Service Instances](#).
3. Choose [Create](#), and enter the following information:
  - *Name* - enter the name of your instance.
  - *Offering Name* - type: **enterprise-messaging**.
  - *Plan Name* - type: **default**.
4. In [Parameters](#), paste the following sample config.json:

```
{  
    "options": {  
        "management": true,  
        "messagingrest": true  
    },  
    "rules": {  
        "topicRules": {  
            "publishFilter": [  
                "${namespace}/*"  
            ],  
            "subscribeFilter": [  
                "${namespace}/*"  
            ]  
        },  
        "queueRules": {  
            "publishFilter": [  
                "${namespace}/*"  
            ],  
            "subscribeFilter": [  
                "${namespace}/*"  
            ]  
        }  
    },  
    "resources": {  
        "units": "10"  
    },  
    "version": "1.1.0",  
    "emname": "<{EVENT_MESH_NAME}>",  
    "namespace": "<{EVENT_MESH_NAMESPACE}>"  
}
```

```
}
```

5. Replace <{EVENT\_MESH\_NAME}> and <{EVENT\_MESH\_NAMESPACE}> with the values you want your SAP Event Mesh instance to have.

#### ⓘ Note

The <{EVENT\_MESH\_NAMESPACE}> must follow the [SAP Event Specification](#); for example, it cannot exceed 64 characters, or begin or end with a dot or hyphen.

6. Go to  [Service Management](#)  [Service Bindings](#) and choose [Create](#).
7. Provide the name of your binding, select the name of your instance from the list, and choose [Create](#).

## Results

You have set up SAP Event Mesh for Kyma Eventing.

## Next Steps

Choose SAP Event Mesh as your backend for Kyma Eventing.

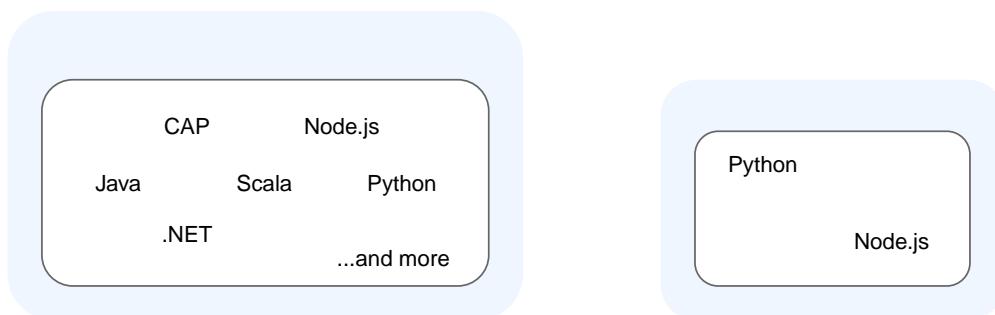
## Related Information

[Choose a Backend for Kyma Eventing \[page 1984\]](#)

## 4.3.8 Available Technology

You can deploy your application in SAP BTP, Kyma runtime as a Docker container.

You can use any technology of choice as long as you can package it as a Docker container. The Docker container can be built using Cloud Native Buildpacks (CNB). It is also possible to build lightweight [Serverless](#) Functions in Python and JavaScript.



Text is not SVG - cannot display

With microservices, you build and maintain the Docker image.

When working with Serverless Functions, you write the code and Kyma takes care of creating and deploying the Docker image.

# 5 Extensions

The extension capabilities of SAP Business Technology Platform (SAP BTP) enables developers to implement loosely coupled extension applications securely, thus implementing additional workflows or modules on top of the existing SAP cloud solution they already have.

## Unified Customer Landscape

The Unified Customer Landscape service provides capabilities for automated extensibility and integration of SAP and third-party systems.

All standard SAP solutions are offered with customizing capabilities. Additionally, customers often have their own requirements for innovative or industry-specific extensions and the extension capabilities of SAP BTP can help them build, deploy, and operate their new functionality.

You can extend standard SAP solutions without disrupting their performance and core processes. When building extension applications, you can also benefit from the automation of the integration between the cloud platform and the extended SAP solutions.

You can also benefit from an automated integration between SAP systems or between SAP systems and a specific service in SAP BTP.

Using the Unified Customer Landscape service, you can maintain your customer landscape. The frontend representation of this service is the [System Landscape](#) page in the SAP BTP cockpit. The dedicated SAP S/4HANA Cloud Extensibility and SAP SuccessFactors Extensibility services are also part of the Unified Customer Landscape capabilities. See [Maintaining Unified Customer Landscape \[page 1990\]](#).

To get a full list of terms related to the extensibility and integration concepts in the Unified Customer Landscape area, see [Extensibility Concepts \[page 1992\]](#).

## Registering and Deregistering Systems

To connect a system with a global account in SAP BTP, you need to have the system listed in the [System](#) page.

[Registering and Deregistering Systems \[page 1997\]](#)

## Extending SAP Solutions

The extension capabilities of SAP BTP provide a standard way for extending SAP solutions and developing event-driven extensions and applications. This framework includes:

- Simplified, standardized and unified extensibility and configuration for the SAP solutions.

- Central catalog per customer for all connected SAP systems where data such as APIs, events, credentials and other is stored. You can benefit from business services and actionable events across end-to-end business processes.

You have the following options to extend your SAP solution using the SAP SuccessFactors Extensibility or SAP S/4HANA Cloud Extensibility services:

- Extensions with automated configurations in the Cloud Foundry runtime: applicable for SAP S/4HANA Cloud, SAP Marketing Cloud, and SAP SuccessFactors.
- Extensions with automated configurations in the Kyma runtime: applicable for SAP S/4HANA Cloud, SAP Marketing Cloud, SAP SuccessFactors, SAP Cloud for Customer, SAP Commerce Cloud, and SAP Field Service Management.

If you have to group the systems of different SAP solutions in the same business case, you can set up the connectivity between all these systems and a global account in SAP BTP in a single formation in the SAP BTP cockpit. See [Including Systems in a Formation \[page 2027\]](#).

See [Extending SAP Solutions \[page 2022\]](#).

## Troubleshooting

If you encounter a problem when extending an SAP S/4HANA Cloud or an SAP SuccessFactors system, go through the following troubleshooting information first:

- [Troubleshooting for SAP S/4HANA Cloud Extensibility Service \[page 2106\]](#)
- [Troubleshooting for SAP SuccessFactors Extensibility Service \[page 2134\]](#)

## Related Information

[SAP BTP Developer's Guide: Extending Existing SAP Solutions Using SAP BTP](#)

## 5.1 Maintaining Unified Customer Landscape

### Use

#### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Unified Customer Landscape is a service that offers customer landscape management capabilities. The frontend representation of this service is the [System Landscape](#) page in the SAP BTP cockpit. The dedicated

*SAP S/4HANA Cloud Extensibility* and *SAP SuccessFactors Extensibility* services are also part of the Unified Customer Landscape capabilities.

## Customer Landscape Discovery

The *System Landscape* page of the SAP BTP cockpit gives you a visual overview of your SAP and third-party systems associated with the given global account or a subaccount. There are different ways to add systems in the **Systems** **Systems** page: manually or automatically. If a system of your solution is associated with your global account or through a subscription in SAP BTP cockpit associated with a given subaccount, it will appear in the list automatically. Otherwise, you have to add your system manually. Systems are added to the list in one of the following ways:

- Auto-Discovered  
An auto-discovered system is a system (associated with the given global account) that has been discovered and added automatically to the list based on information of the existing system landscape. Any SAP system of the supported system types that is associated with the same customer ID, with which your global account in SAP BTP is associated, will be added automatically in the system landscape list.
- Subaccount/<my-subaccount>  
Specifies that the system has been added through a subscription in SAP BTP cockpit associated with a given subaccount. The subscription has been discovered and added automatically through the subaccount.
- Manually-Added  
Specifies that the system has been added to the list manually by the global account administrator, using the *Add System* button and completing the wizard. The system has been associated with the global account in SAP BTP.  
See [Registering and Deregistering Systems \[page 1997\]](#).

## Integration and Extensibility

Unified Customer Landscape also helps you integrate and extend your SAP S/4HANA Cloud, SAP Ariba, SAP SuccessFactors, and other SAP and third-party systems in one single experience.

In the SAP BTP cockpit, you get a comprehensive overview of all your systems associated with your customer ID. These systems can be registered or auto discovered. They are conveniently listed as a record in the *Systems* list. Moreover, Unified Customer Landscape lets you integrate one or more systems in a common business case by including these systems in a formation.

See:

- [Including Systems in a Formation \[page 2027\]](#)
- [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#)
- [Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment \[page 2108\]](#)
- [Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 2112\]](#)
- [Extending SAP Customer Experience Products in the Kyma Environment \[page 2136\]](#)

## 5.2 Extensibility Concepts

When implementing your extensibility scenario, there are a couple of concepts you need to understand to be able to benefit from the functionality that SAP BTP offers.

### Systems

A system is a specific instance of an SAP or third-party solution that is manually added or auto discovered and is listed in the  [System Landscape](#)  page in the SAP BTP cockpit.

When you want to add functionality to your SAP or third-party solution, you start by developing an extension application and deploying it in SAP BTP. Then, to allow the application to access the SAP or third-party solution, you add the system of this solution to the [Systems](#) page of the SAP BTP cockpit. If the system is one of the following types, you also have to register it using a registration token:

- SAP S/4HANA Cloud
- SAP Marketing Cloud
- SAP SuccessFactors

### System Details

#### Status

The registration process can have the following status values:

- No status  
The system has been added as a record to the list. You can get a registration token for it. However, the registration process on the corresponding solution's system side has not been performed or completed yet.
- *Registered*  
The registration token has been used and the automated registration process has been completed successfully. Now, the system can be used in extensibility business scenarios.
- *Error while Registering*  
The registration has failed.
- *Deregistering*  
A deregistration process has started. As a result, the connectivity between the system and SAP BTP is disabled and extension scenarios cannot be established. The system remains in the system landscape list and you can register it again later on.
- *Error while Deregistering*  
The deregistration has failed.
- *Removing*  
A system removal process has started in the SAP BTP cockpit. You can remove a system from the list only when this system:
  - Hasn't been registered yet
  - Has been deregistered first
  - Has been manually added

Then, you can remove the system from the [Systems](#) list completely. To register the system again, first, you must add it to the list anew, and then, initiate the registration procedure.

- [Error while Removing](#)

The system removal has failed.

## Consumption Bundles, APIs, and Events

Consumption bundles group logically APIs and events intended for communication with the SAP or third-party system. This grouping means that the consumption of the APIs or the events can happen by using the same set of credentials later on. When you register a system, you can also preview the number and the content of consumption bundles that are exposed for communication with the given system.

After you add your third-party systems, you can specify the set of APIs and events in consumption bundles. A consumption bundle organizes a set of related APIs and events into a single group for consumption purposes and expresses information about how the APIs and events that it contains can be accessed. All APIs and events that are part of the same consumption bundle need to be accessible through the same set of credentials. You add the consumption bundles in [System Details](#) of the respective third-party system.

You can explore the consumption bundles, the APIs, and the events configured for your system in [System Details](#). To do that, go to the respective tab: [Consumption Bundles](#), [APIs](#), or [Events](#).

## URL

You can use the URL as an entry point to the corresponding system. Optionally, you can use it to identify duplicate systems added to the list.

## Discovery Mechanisms

There are different ways to add systems to the [Systems](#) page in the SAP BTP cockpit: manually or automatically. If a system of your solution is associated with your global account or through a subscription in SAP BTP cockpit associated with a given subaccount, it will appear in the list automatically. Otherwise, you have to add your system manually. Systems are added to the list in one of the following ways:

### Auto-Discovered

An auto-discovered system is a system (associated with the given global account) that has been discovered and added automatically to the list based on information of the existing system landscape. Any SAP system of the supported system types that is associated with the same customer ID, with which your global account in SAP BTP is associated, will be added automatically in the system landscape list.

#### ⓘ Note

If a given SAP system is missing on the [Systems](#) page, it may be associated with a different customer ID on the SAP BTP global account you are working in. In this case, you need to add the system manually, and then, register it.

### Subaccount/<my-subaccount>

Specifies that the system has been added through a subscription in SAP BTP cockpit associated with a given subaccount. The subscription has been discovered and added automatically through the subaccount.

### Manually-Added

Specifies that the system has been added to the list manually by the global account administrator, using the [Add System](#) button and completing the wizard. The system has been associated with the global account in SAP BTP.

## **Business Type**

When you have an auto-discovered SAP system, you can find information in the respective system details whether this system is used for test or production purposes for example. This information is loaded automatically and is presented in the *Business Type* field.

## **Actions**

These are the things you can do with a system:

### **Adding**

Adding a system to the list in the [Systems](#) page is just the first step of the system registration process. When you have only added a system, the system is not yet registered in the global account in SAP BTP. It appears in the system landscape list as a record with empty (or initialized) status. That is, the required configuration on the system side has not been performed, and therefore, the newly added system cannot exchange or expose its technical details, metadata, APIs, or events. Only when the registration process is complete and the system is registered with SAP BTP, it can exchange the relevant information and enable the extension scenario.

### **Registering**

When registering a system, the required configuration on the system side has been performed, and therefore, the newly added system can exchange or expose its technical details, metadata, APIs, or events. Only when the registration process is complete and the system is registered with SAP BTP, it can exchange the relevant information and enable the extension scenario.

The registration of the newly added systems is based on a registration token. The token is used for the pairing of the system and the corresponding global account. After you add a system, you can get the token in the SAP BTP cockpit. Then, you can use it to configure the integration on the corresponding system side. By using the registration token on the system side and registering the system, the system administrator allows the integration of the system with SAP BTP.

The system gets a *Registered* status, only when a token is issued and the registration is complete on the corresponding system side. In general, the *Registered* status means that the communication between the system and SAP BTP has been established. However, depending on the system and its requirements, additional configuration might be needed for the enablement of a fully functional extension scenario. The additional configuration, depending on the system type, is outlined in the corresponding documents listed in the *Related Information* section. See [Registering and Deregistering Systems \[page 1997\]](#).

### **Specifics When Registering SAP S/4HANA Cloud Systems**

When you register an SAP S/4HANA Cloud system, you can automate some of the required configuration steps at a later point using communication scenario groups. The communication scenario groups are entities that are part of the registration token that you use to register your SAP S/4HANA Cloud systems and are associated with one or more communication scenarios in SAP S/4HANA Cloud.

For example, you can use the Event Mesh communication scenario group when you get the registration token for the SAP S/4HANA Cloud system. This allows the automatic enablement of the communication scenario SAP\_COM\_0892 after the corresponding system is added to the formation of type *Eventing Between SAP Cloud Systems*.

You can configure the communication scenario groups when registering an SAP S/4HANA Cloud system in the SAP BTP cockpit, in [System Landscape](#) [Systems](#). See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

## **Merging**

You have added your SAP systems manually at a given point in time and you have registered them. Then, you notice that you have the exact same systems in the list in the *Systems* page only this time they are auto-discovered and are not registered. To avoid duplicates and streamline the list of systems, you can merge the auto-discovered SAP systems with your manually added systems. The logic automatically detects duplicates based on their URL and system type. Note that you can only merge auto-discovered SAP systems without assigned status in a target system that has already been registered. That is, you cannot merge registered auto-discovered systems. See [Merging SAP Systems \[page 2021\]](#).

### Deregistering

Deregistering a system means that the connectivity between this system and the global account is disabled and extension scenarios cannot be established. You can deregister a system from the *Actions* column, or from the *System Details* page that you access when selecting the system from the *System* list. See [Deregistering or Removing a System \[page 2019\]](#).

### Removing

Removing a system means that this system is no longer part of the system landscape list. To remove a system from the list, first you have to deregister it. You can remove a system from the *Actions* column, or from the *System Details* page that you access when selecting the system from the *System* list. See [Deregistering or Removing a System \[page 2019\]](#).

You can see the discovery mechanism and access all the actions related to the systems listed in the SAP BTP cockpit, in ► *System Landscape* ► *Systems* ▶. See [Registering and Deregistering Systems \[page 1997\]](#).

## Formations

A formation is a logical grouping of SAP or third-party systems that can be extended in a business scenario. Formations allow you to combine SAP or third-party solution systems and a subaccount in SAP BTP to simplify the connectivity setup and to provide a unified view of all components required for the implementation of your extension scenario. To create a fully functional formation, you can use a two-step wizard. At the first step, you specify a custom formation name and assign a subaccount to it. At the second step, you can include an SAP or third-party solution system in the formation. You do this configuration once and you can change it anytime.

### ⓘ Note

You can assign a subaccount to the formation during the formation creation only. While you can include or exclude systems in the formation anytime, you cannot unassign or reassign subaccounts later on. Instead, you must recreate the formation.

Extension business cases often involve extending several SAP or third-party solutions at a time. For example, for a given business case you might need to extend the functionality or the UI as follows:

- An SAP SuccessFactors system, and an SAP S/4HANA Cloud system. First, you need to configure the connectivity of each of these systems to Cloud Foundry, Kyma, or both environments. Extension applications of both solutions are part of the same business need.
- An SAP Commerce Cloud system, and an SAP S/4HANA Cloud system. Again, you first configure the connectivity of each of these systems to the Kyma environment.
- A single system of the supported SAP or third-party solutions.
- SAP or third-party systems that expose event data, which can be shared and exchanged with the systems included in the formation.

When creating a formation in the  [System Landscape](#)  [Formations](#) page in the SAP BTP cockpit, you include the systems of the different SAP or third-party solutions you want to extend. If your business case features more than one system, you can use the corresponding button to include additional systems in the formation. You can start the dialog as many times and add systems to your formation as you want.

## Systems Details in a Formation

### Consumption Bundle APIs

A consumption bundle can group logically a set of APIs for communication with the SAP or third-party system.

### Consumption Bundle Events

A consumption bundle can group logically a set of events for communication with the SAP or third-party system.

### Types

The formation type defines the use case. Therefore, depending on the use case, you have different formation types. To get a list of the available formation types, see [Including Systems in a Formation \[page 2027\]](#).

## Formation Status

A formation can have the following status values:

- No status
- *Action Required*  
The formation has been created but you cannot use it productively yet.  
Based on the formation type and the subaccount that you specified, it might require an SAP BTP Kyma environment instance or an SAP Business Application Studio subscription. Although the SAP BTP cockpit allows you to create such a formation, to enable and make use of it, you must also create the respective instance or subscription.
- *Synchronizing*  
Systems that are included in the formation are currently synchronizing in the background.
- *Error*  
An error occurred while some of the systems that are included in the formation were synchronizing in the background.

## Actions

### Creating Formations

Create a logical grouping of the systems that you want to extend.

### Including/Excluding Systems in a Formation

Include additional systems to a given formation or exclude already included systems from a formation.

If your business case features more than one system, you can include additional systems in the formation. You can include as many systems as you want to your formation.

### Resynchronizing Systems in a Formation

When you include systems in a formation, these systems are synchronized in the background. If an error occurs, you can resynchronize these systems to restart the synchronization process. To do that, choose *Resynchronize* for the particular formation. You will have this action available only for formations that are in status *Error*.

### Resetting and Resynchronizing Systems in a Formation

If you want to reconfigure the systems in a formation from scratch without excluding them, you have to choose [Reset and Resynchronize](#). You will have this action available only for formations that have no status in SAP BTP cockpit their formation type is Side-by-Side Extensibility with Kyma.

### Deleting Formations

Detach the systems, unassign the subaccount, and delete the formation at one go.

When you start a formation deletion process, first the systems are excluded from the formation, then, the subaccount is unassigned, and last, the formation is deleted from the list completely.

You can create and configure formations in the SAP BTP cockpit, in [System Landscape](#) [Formations](#). See [Including Systems in a Formation \[page 2027\]](#).

## 5.3 Registering and Deregistering Systems

To connect an SAP system with a global account in SAP BTP, you need to register the system. You can also add and work with a third-party systems.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator of the global account where you want to register your SAP system. See [Working with Role Collections \[page 2274\]](#).
- Before adding your system, make sure it's not already auto-discovered and listed automatically in the [System Landscape](#) [Systems](#) page.

### Registering SAP Systems

#### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

For some system types, such as SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, adding a system to the list in the [Systems](#) page is just the first step of the system registration process. When you have only added a system, the system is not yet registered in the SAP BTP global account. That is, the required configuration on the system side has not been performed, and therefore, the newly added system cannot exchange or expose its technical details, metadata, APIs, or events. Only when the registration process is complete and the system is registered with SAP BTP, it can exchange the relevant information and enable the extension scenario.

If your system is associated with the given global account, this system is discovered and added automatically to the [Systems](#) page based on information of the existing system landscape. Any SAP system that is associated with the same customer ID, with which your global account in SAP BTP is associated, will be auto-discovered.

### Note

If a given SAP system is missing on the [Systems](#) page, it may be associated with a different customer ID on the SAP BTP global account you are working in. In this case, you need to add the system manually, and then, register it.

### Note

You cannot migrate the registered SAP systems between global accounts.

If you want to start using another global account, you will have to register your SAP systems again.

For systems of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, the registration of the newly added systems is based on a registration token. The registration token is used for the pairing of the system and the corresponding global account.

After you add a system, you can get the token in the SAP BTP cockpit. Then, you can use it to configure the integration on the corresponding system side. By using the registration token on the system side and registering the system, the system administrator allows the integration of the system with SAP BTP.

When you add an SAP system of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, it appears in the system landscape list as a record with empty (or initialized) status. The system gets a [Registered](#) status, only when a token is issued and the registration is complete on the corresponding system side. In general, the [Registered](#) status means that the communication between the system and SAP BTP has been established. However, depending on the system and its requirements, additional configuration might be needed for the enablement of a fully functional extension scenario. The additional configuration, depending on the system type, is outlined in the corresponding documents listed in the [Related Information](#) section.

### Note

When registering SAP systems of the same type, you can have up to 1000 tokens per global account ready to be used. Tokens that are already used to register an SAP system are not included in this number.

This means that you cannot have more than 1000 systems in the Systems list of the same type with an empty status and generated token that is not used yet.

For system types different than SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, when adding a system in the [Systems](#) page, this system is registered directly but no status is displayed.

The registration process has the following states displayed in the cockpit:

- No status displayed in the [Status](#) column - the registration token for an SAP system of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud or SAP systems from the SAP Customer Experience portfolio has been created but the registration on the respective SAP system side has not been performed or completed yet. For systems with type other than SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio are registered when added to the [Systems](#) page but no status has been displayed. Only systems that require a registration token to be registered have the status [Registered](#) when this token has been used.
- [Registered](#) - the registration token for systems of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio has been used and the automated registration process has been successfully completed. The systems of these types can be

assigned to a formation on the *Formations* page in the cockpit. All other system types are registered without a registration token and can be assigned to a formation.

- *Error while Registering* - the registration has failed.

#### ⓘ Note

If a system is in status *Error while Registering*, delete it and register it again.

If the registration error persists, report a case in one of the following components depending on the system type:

- SAP SuccessFactors - BC-NEO-EXT-SF
- SAP S/4HANA Cloud - BC-NEO-EXT-S4C
- All other system types - BC-CP-MP

#### ⓘ Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. In the cockpit, navigate to your global account, and then choose **System Landscape** **Systems**.
2. On the *Systems* page, choose *Add System*.

#### ⓘ Note

Before adding your system, make sure it's not already auto-discovered and listed automatically in the *Systems* page. Some of the SAP systems have additional configurations described in dedicated procedures. See:

- [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#)
- [Register an SAP Marketing Cloud System in a Global Account in SAP BTP \[page 2006\]](#)
- [Register an SAP SuccessFactors System in a Global Account in SAP BTP \[page 2009\]](#)
- [Register an SAP Customer Experience System \[page 2014\]](#)

3. In the *Add System* wizard:

1. Enter a name for the system you want to register.

#### ⓘ Note

Use only printable ASCII characters.

#### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, `<my system>-commerce-cloud`. This helps you identify the system type when assigning systems to a formation.

2. In the *Type* dropdown list, select the system type.
3. Choose *Add*.
4. For systems of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, choose *Get Token*.

The system generates the registration token.

Systems of type different than SAP SuccessFactors, SAP S/4HANA Cloud and SAP systems from the SAP Customer Experience portfolio don't need a registration token, they are automatically registered when added to the [Systems](#) page. Their status is empty but you can consider them as registered.

5. Copy the registration token. You need the token to complete the integration on the respective SAP solution system side.
6. Close the wizard.

## Result

The system has been added as a record to the list on the [Systems](#) page in the SAP BTP cockpit. For systems of type SAP SuccessFactors, SAP S/4HANA Cloud and SAP systems from the SAP Customer Experience portfolio, a registration token for connecting the corresponding SAP solution with the global account in SAP BTP has been generated. Systems of type other than SAP SuccessFactors, SAP S/4HANA Cloud and SAP systems from the SAP Customer Experience portfolio, are automatically registered, no additional configurations are required.

### Note

Registration tokens have different validity periods that depend on the system type. For more information about token expiry, see the corresponding documentation at the [Related Links](#) section.

## Next Steps

- For SAP systems of type SAP SuccessFactors, SAP S/4HANA Cloud, SAP Marketing Cloud and SAP systems from the SAP Customer Experience portfolio, use the registration token to complete the registration on the respective SAP system side. See:
  - [Trigger the Registration in the SAP S/4HANA Cloud Tenant \[page 2005\]](#)
  - [Trigger the Registration in the SAP Marketing Cloud Tenant \[page 2009\]](#)
  - [Register an SAP SuccessFactors System in a Global Account in SAP BTP \[page 2009\]](#)
  - [Register an SAP Customer Experience System \[page 2014\]](#)
- You can assign the system to a formation on the [Formations](#) page, as follows:
  - SAP systems of type [SAP Commerce Cloud](#), [SAP Cloud for Customer](#), and [SAP Field Service Management](#) can be assigned to a formation directly before the integration is complete on the respective SAP Customer Experience system side. However, to enable the API access, you first need to complete the integration.
  - For systems of type [SAP S/4HANA Cloud](#), [SAP Marketing Cloud](#) and [SAP SuccessFactors](#), you first need to configure the integration on the respective SAP system side.
  - For systems of type other than [SAP S/4HANA Cloud](#), [SAP Marketing Cloud](#) and [SAP SuccessFactors](#) and SAP systems from the SAP Customer Experience portfolio, after adding these systems in the [Systems](#) list, you can directly include them in a formation.
- See [Including Systems in a Formation \[page 2027\]](#).
- If you no longer need it, you can deregister or remove the system depending on its status. See [Deregistering or Removing a System \[page 2019\]](#).

## Registering Third-Party Systems

You add a third-party system to the list in the  [System Landscape](#)  [Systems](#)  page. At this point you provide all the required details for this system: its type, provider, URL, and system ID. For third-party systems, this

completes the registration process and you have your third-party system registered with SAP BTP. Even though the third-party system is registered directly, no status is displayed.

See [Registering a Third-Party System \[page 2016\]](#).

## Deregistering Systems

Deregistering an SAP or third-party system means that the connectivity between this system and the global account is disabled and extension scenarios cannot be established, while removing a system means that this system is no longer part of the system landscape list. You can deregister or remove a system from the *Actions* column, or from the *System Details* page that you access when selecting the system from the system landscape list.

See [Deregistering or Removing a System \[page 2019\]](#).

## Merging Systems

If you have already added these systems manually at a given point in time and you have registered them, the auto-discovered systems might create duplicate entries in the list. To avoid duplicates and streamline the list of systems, you can merge automatically the auto-discovered SAP systems with your manually added systems.

See [Merging SAP Systems \[page 2021\]](#).

## Related Information

[Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#)

[Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment \[page 2108\]](#)

[Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 2112\]](#)

[Extending SAP Customer Experience Products in the Kyma Environment \[page 2136\]](#)

[Registering a Third-Party System \[page 2016\]](#)

## 5.3.1 Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP

To connect an SAP S/4HANA Cloud Public Edition system with a global account in SAP BTP, you need to register the system in the corresponding global account.

### Prerequisites

- See [Registering and Deregistering Systems \[page 1997\]](#).
- You are an administrator of the global account where you want to register your SAP S/4HANA Cloud system.
- To configure the integration on the SAP S/4HANA Cloud system side, you need to be an administrator of the SAP S/4HANA Cloud tenant.

### Context

#### ⓘ Note

This documentation refers to SAP S/4HANA Cloud Public Edition. See [Introduction to the Universe of SAP S/4HANA Cloud Public Edition](#).

The registration process is based on a registration token that is used for the pairing of the SAP S/4HANA Cloud system and the corresponding global account. You create the token in the global account, and then the tenant administrator of the respective SAP S/4HANA Cloud system uses the token to start the automated registration process on the SAP S/4HANA Cloud system side.

#### ⓘ Note

When registering SAP S/4HANA Cloud systems, you can have up to 1000 tokens per global account ready to be used. Tokens that are already used to register an SAP S/4HANA Cloud system are not included in this number.

This means that you cannot have more than 1000 systems in the [Systems](#) page of type SAP S/4HANA Cloud with an empty status and generated token that is not used yet.

#### ⓘ Note

You cannot migrate the registered SAP S/4HANA Cloud systems between global accounts.

If you want to start using another global account, you will have to register your SAP S/4HANA Cloud systems again.

The registration process has the following states displayed in the cockpit:

- No status displayed in the [Status](#) column - the registration token for an SAP system has been created but the registration on the respective SAP solution system side has not been performed or completed.

- *Registered* - the registration token has been used and the automated registration process has been successfully completed. The system can be assigned to a formation on the *Formations* page in the cockpit.
  - *Error while Registering* - the registration has failed. Remove the system and then add it to the *Systems* list and try to register it again
  - *Deregistering* - a deregistration process has started in the SAP BTP cockpit. As a result, the connection between the SAP solution system and the global account in SAP BTP is removed. The system remains in the list and you can register it again later on.
- Once a system is registered, you can deregister it only after removing it from all entitlement configurations and formations it takes part in.

#### Note

You will not be able to deregister a system if its status is one of the following:

- *Error while Registering*
- *Deregistering*

- *Error while Deregistering* - the deregistration has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-S4C component.
- *Removing* - a system removal process has started in the SAP BTP cockpit. As a result, the link between the SAP solution and SAP BTP is destroyed and the system is removed from the list. To register the system again, first you must add it to the list anew, and then initiate the registration procedure.

Once a system is registered, you can only remove it if you first deregister it.

#### Note

You will not be able to remove a system if its status is one of the following:

- *Registered*  
You first need to deregister the system.
- *Deregistering*
- *Error while Deregistering*  
Try to deregister the system again.

- *Error while Removing* - the system removal has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-S4C component.

#### Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. In the cockpit, navigate to your global account, and then choose   *System Landscape*  *Systems*.
2. On the *Systems* page, choose *Add System*.
  - a. Enter a name for the system you want to register.

### ⓘ Note

Use only printable ASCII characters.

### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, <mysystem>-S/4HANA-cloud. This helps you identify the system type when assigning systems to a formation.

- b. In the *Type* dropdown list, select *SAP S/4HANA Cloud*.
- c. Choose *Add*.
- d. In the *Communication Scenario Groups* dropdown menu, select one of the following options:
  - *All Communication Scenarios*: to create any of the communication scenarios in SAP S/4HANA Cloud.
  - *Eventing Between SAP Cloud Systems*: to allow the automatic enablement of the SAP\_COM\_0892 communication scenario in SAP S/4HANA Cloud.
  - *Integration with SAP Ariba Buying*: to allow the automatic enablement of the SAP\_COM\_0545 and SAP\_COM\_0A00 communication scenarios in SAP S/4HANA Cloud.
  - *Integration with SAP Start*: to allow the automatic enablement of the SAP\_COM\_0647 communication scenario in SAP S/4HANA Cloud.
- e. Choose *Get Token*.

The system generates the registration token.

- f. Copy the registration token and send it to the tenant administrator for the respective SAP S/4HANA Cloud system. You need it for configuring the integration on the extended SAP S/4HANA Cloud system side.

You can also get the registration token later, once the system appears in the list on the *Systems* page.

The registration token is valid for 7 days after it has been generated. When a token is not used within its validity period, it is no longer valid and cannot be used for registering an SAP S/4HANA Cloud system. If the validity of the token expires before you use it to configure the integration on the SAP S/4HANA Cloud system side and complete the registration, you need to get a new token. You can then copy it and use it to complete the registration.

### ⓘ Note

A registration token can be used only once, for registering a single SAP S/4HANA Cloud system.

- g. Close the wizard.

The SAP S/4HANA Cloud system appears in the list of systems on the *Systems* page. Its *Status* field is empty because the registration process is not yet completed.

3. Start the automated registration process on the SAP S/4HANA Cloud system side. To do so, proceed as described in as described in [Trigger the Registration in the SAP S/4HANA Cloud Tenant \[page 2005\]](#).

### ⓘ Note

You can register a system with the same name only once per global account. Once you have started a registration process for a system with a specified name you can no longer register a system with the same name and connect it with the same global account, unless you delete the corresponding extension in the [Maintain Extensions on SAP BTP](#) in the SAP S/4HANA Cloud tenant. If the registration

If the registration process fails, you need to delete the failed extension from the SAP S/4HANA Cloud tenant and create a new integration token in the cockpit for the corresponding system to be able to start the automated registration process again.

4. Check the status of the registration process. To do so, in the cockpit navigate to your global account, and on the [Systems](#) page, check if the status of the SAP S/4HANA Cloud system has changed to *Registered*.

If you are already on the [Systems](#) page, refresh the page to check if the status has changed.

## Next Steps

[Trigger the Registration in the SAP S/4HANA Cloud Tenant \[page 2005\]](#)

## Related Information

[Registering and Deregistering Systems \[page 1997\]](#)

[Deregistering or Removing a System \[page 2019\]](#)

### 5.3.1.1 Trigger the Registration in the SAP S/4HANA Cloud Tenant

Use this procedure to trigger the registration process for an SAP S/4HANA Cloud system that you want to pair with your global account in SAP BTP.

#### Prerequisites

You are an SAP S/4HANA Cloud tenant administrator.

#### Procedure

1. Log on to the SAP S/4HANA Cloud tenant, go to [Home](#), [Communication Management](#) tab and then choose the [Maintain Extensions on SAP BTP](#) tile.
2. In the [Maintain Extensions on SAP BTP](#) screen, in the [Extensions](#) section, choose [New](#).
3. In the [Integration Token](#) field, enter the content of the registration token from the SAP BTP cockpit.
4. Choose [Save](#). A new entry in the table appears with status *Enabling*.
5. After the integration has finished successfully, you can refresh the table.

The status of the integration should have changed to *Enabled*.

## 5.3.2 Register an SAP Marketing Cloud System in a Global Account in SAP BTP

To connect an SAP Marketing Cloud system with a global account in SAP BTP, you need to register the system in the corresponding global account.

### Prerequisites

- See [Registering and Deregistering Systems \[page 1997\]](#)
- You are an administrator of the global account where you want to register your SAP Marketing Cloud system.
- To configure the integration on the SAP Marketing Cloud system side, you need to be an administrator of the SAP Marketing Cloud tenant.

### Context

The registration process is based on a registration token that is used for the pairing of the SAP Marketing Cloud system and the corresponding global account. You create the token in the global account, and then the tenant administrator of the respective SAP Marketing Cloud system uses the token to start the automated registration process on the SAP Marketing Cloud system side.

#### ⓘ Note

When registering SAP Marketing Cloud systems, you can have up to 1000 tokens per global account ready to be used. Tokens that are already used to register an SAP Marketing Cloud system are not included in this number.

This means that you cannot have more than 1000 systems in the [Systems](#) page of type SAP Marketing Cloud with an empty status and generated token that is not used yet.

The registration process has the following states displayed in the cockpit:

- No status displayed in the *Status* column - the registration token for an SAP system has been created but the registration on the respective SAP solution system side has not been performed or completed.
- *Registered* - the registration token has been used and the automated registration process has been successfully completed. The system can be assigned to a formation on the [Formations](#) page in the cockpit.
- *Error while Registering* - the registration has failed. Remove the system and then add it to the [Systems](#) list and try to register it again
- *Deregistering* - a deregistration process has started in the SAP BTP cockpit. As a result, the connection between the SAP solution system and the global account in SAP BTP is removed. The system remains in the list and you can register it again later on.

Once a system is registered, you can deregister it only after removing it from all entitlement configurations and formations it takes part in.

#### ⓘ Note

You will not be able to deregister a system if its status is one of the following:

- *Error while Registering*
- *Deregistering*

- *Error while Deregistering* - the deregistration has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-S4C component..
- *Removing* - a system removal process has started in the SAP BTP cockpit. As a result, the link between the SAP solution and SAP BTP is destroyed and the system is removed from the list. To register the system again, first you must add it to the list anew, and then initiate the registration procedure.

Once a system is registered, you can only remove it if you first deregister it.

#### ⓘ Note

You will not be able to remove a system if its status is one of the following:

- *Registered*  
You first need to deregister the system.
- *Deregistering*
- *Error while Deregistering*

- *Error while Removing* - the system removal has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-S4C component.

#### ⓘ Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

#### ⓘ Note

You cannot migrate the registered SAP Marketing Cloud systems between global accounts.

If you want to start using another global account, you will have to register your SAP Marketing Cloud systems again.

## Procedure

1. In the cockpit, navigate to your global account, and then choose **System Landscape** **Systems**.
2. On the **Systems** page, choose **Add System**.
  - a. Enter a name for the system you want to register.

#### ⓘ Note

Use only printable ASCII characters.

### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, `<mysystem>-S/4HANA-cloud`. This helps you identify the system type when assigning systems to a formation.

- b. In the *Type* dropdown list, select *SAP Marketing Cloud*.
- c. Choose *Add*.
- d. Choose *Get Token*.

The system generates the registration token.

- e. Copy the registration token and send it to the tenant administrator for the respective SAP Marketing Cloud system. You need it for configuring the integration on the extended SAP Marketing Cloud system side.

You can also get the registration token later, once the system appears in the list on the *Systems* page.

The registration token is valid for 7 days after it has been generated. When a token is not used within its validity period, it is no longer valid and cannot be used for registering an SAP Marketing Cloud system. If the validity of the token expires before you use it to configure the integration on the SAP Marketing Cloud system side and complete the registration, you need to get a new token. You can then copy it and use it to complete the registration.

### ⓘ Note

A registration token can be used only once, for registering a single SAP Marketing Cloud system.

- f. Close the wizard.

The SAP Marketing Cloud system appears in the list of systems on the *Systems* page. Its *Status* field is empty because the registration process is not yet completed.

3. Start the automated registration process on the SAP Marketing Cloud system side. To do so, proceed as described in as described in [Trigger the Registration in the SAP Marketing Cloud Tenant \[page 2009\]](#).

### ⓘ Note

You can register a system with the same name only once per global account. Once you have started a registration process for a system with a specified name you can no longer register a system with the same name and connect it with the same global account, unless you delete the corresponding extension in the [Maintain Extensions on SAP BTP](#) in the SAP Marketing Cloud tenant. If the registration process fails, you need to delete the failed extension from the SAP Marketing Cloud tenant and create a new registration token in the cockpit for the corresponding system to be able to start the automated registration process again.

4. Check the status of the registration process. To do so, in the cockpit navigate to your global account, and on the *Systems* page, check if the status of the SAP Marketing Cloud system has changed to *Registered*.

If you are already on the *Systems* page, refresh the page to check if the status has changed.

## Next Steps

[Trigger the Registration in the SAP Marketing Cloud Tenant \[page 2009\]](#)

## Related Information

[Registering and Deregistering Systems \[page 1997\]](#)

[Deregistering or Removing a System \[page 2019\]](#)

### 5.3.2.1 Trigger the Registration in the SAP Marketing Cloud Tenant

Use this procedure to trigger the registration process for an SAP Marketing Cloud system that you want to pair with your global account in SAP BTP.

#### Prerequisites

You are an SAP Marketing Cloud tenant administrator.

#### Procedure

1. Log on to the SAP Marketing Cloud tenant, go to *Home*, *Communication Management* tab and then choose the *Maintain Extensions on SAP BTP* tile.
2. In the *Maintain Extensions on SAP BTP* screen, in the *Extensions* section, choose *New*.
3. In the *Integration Token* field, enter the content of the registration token from the SAP BTP cockpit.
4. Choose *Save*. A new entry in the table appears with status *Enabling*.
5. After the integration has finished successfully, you can refresh the table.

The status of the integration should have changed to *Enabled*.

### 5.3.3 Register an SAP SuccessFactors System in a Global Account in SAP BTP

To connect an SAP SuccessFactors system with a global account in SAP BTP, you need to register the system in the corresponding global account.

#### Prerequisites

- See [Registering and Deregistering Systems \[page 1997\]](#).

- You are an administrator of the global account where you want to register your SAP SuccessFactors system.
- You have a dedicated SAP SuccessFactors company instance.
- You have a user with permissions to access [Extension Center](#) in SAP SuccessFactors Admin Center that include:
  - [Admin Access to MDF OData API](#) permission from the [Metadata Framework](#) category
  - [Create Integration with SAP BTP](#) permission from the [Manage Extensions on SAP BTP](#) category

To assign these permissions to your user, you might need the help of an administrator.

To get a user with the respective permissions, follow these steps:

1. Make sure you have an access to the SAP SuccessFactors Admin Center. See [Permission to Access Admin Center](#).
2. Add your user to an already existing group or create a new dedicated group and add your user. See [Creating Dynamic Permission Groups](#).
3. Use an already existing role or create a new dedicated role. See [Creating Permission Roles](#).
4. Assign the respective permissions to your role. See [Assigning Permissions to a Role](#).
5. Add your role to the dedicated group. See [Assigning Roles to Groups](#).
6. To fully activate the permissions, log out and log in again to the SAP SuccessFactors company instance.

## Context

The registration process is based on a registration token that is used for the pairing of the SAP SuccessFactors company and the corresponding global account in SAP BTP. You create the token in the global account, and then start the automated registration process on the SAP SuccessFactors company side using this token.

### ⓘ Note

When registering SAP SuccessFactors systems, you can have up to 1000 tokens per global account ready to be used. Tokens that are already used to register an SAP SuccessFactors system are not included in this number.

This means that you cannot have more than 1000 systems in the [Systems](#) list of type SAP SuccessFactors with an empty status and generated token that is not used yet.

The registration process has the following states displayed in the cockpit:

- No status displayed in the [Status](#) column - the registration token for an SAP system has been created but the registration on the respective SAP solution system side has not been performed or completed.
- [Registered](#) - the registration token has been used and the automated registration process has been successfully completed. The system can be assigned to a formation on the [Formations](#) page in the cockpit.
- [Error while Registering](#) - the registration has failed. Remove the system and then add it to the [Systems](#) list and try to register it again
- [Deregistering](#) - a deregistration process has started in the SAP BTP cockpit. As a result, the connection between the SAP solution system and the global account in SAP BTP is removed. The system remains in the list and you can register it again later on.

Once a system is registered, you can deregister it only after removing it from all entitlement configurations and formations it takes part in.

- *Error while Deregistering* - the deregistration has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-SF component.
  - *Removing* - a system removal process has started in the SAP BTP cockpit. As a result, the link between the SAP solution and SAP BTP is destroyed and the system is removed from the list. To register the system again, first you must add it to the list anew, and then initiate the registration procedure.
- Once a system is registered, you can only remove it if you first deregister it.

#### ⓘ Note

You will not be able to remove a system if its status is one of the following:

- *Registered*  
You first need to deregister the system.
- *Deregistering*
- *Error while Deregistering*  
Try to deregister the system again.
- *Error while Removing* - the system removal has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-NEO-EXT-SF component.

#### ⓘ Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. In the SAP BTP cockpit, navigate to your global account, and then choose  [System Landscape](#) .
2. On the [Systems](#) page, choose [Add System](#).
  - a. Enter a name for the system you want to register.

#### ⓘ Note

Use only printable ASCII characters.

#### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, `<mysystem>-SuccessFactors`. This helps you identify the system type when assigning systems to a formation.

- b. In the [Type](#) dropdown list, select SAP SuccessFactors.
  - c. Choose [Add](#).
  - d. Choose [Get Token](#).
- The system generates the registration token.
- e. Copy the registration token and send it to the tenant administrator for the respective SAP SuccessFactors system. You need it for configuring the integration on the extended SAP SuccessFactors system side.

You can also get the registration token later, once the system appears in the list on the [Systems](#) page.

The registration token is valid for 7 days after it has been generated. When a token is not used within its validity period, it is no longer valid and cannot be used for registering an SAP SuccessFactors system. If the validity of the token expires before you use it to configure the integration on the SAP SuccessFactors system side and complete the registration, you need to get a new token. You can then copy it and use it to complete the registration.

#### ⓘ Note

A token can be used only once, for registering a single SAP SuccessFactors system.

- f. Close the wizard.

The SAP SuccessFactors system appears in the list of systems on the [Systems](#) page. Its [Status](#) field is empty because the registration process is not yet completed.

3. Start the automated integration process on the SAP SuccessFactors company side:

#### ⓘ Note

If you do not have permissions to access the Extension Center for the corresponding SAP SuccessFactors system, you need to send the registration token to a user with such permissions who will configure the integration on the SAP SuccessFactors system side. For the required permissions, check the prerequisites.

1. In SAP SuccessFactors [Admin Center](#), navigate to [Extension Center](#).
2. On the [Extensions on SAP BTP](#) tab page, navigate to the [Add Integration with SAP BTP](#) screen area, and paste the registration token in the [Integration Token](#) input field.
3. Choose [Add](#).

The system appears in the integration list in the [Multi-Cloud Environment](#) screen area, and the status of the integration is displayed in the [Integration Status](#) column. To refresh the status of the process, choose the [Check Status](#) icon. Wait for the integration to finish.

The screenshot shows the SAP SuccessFactors Admin Center Extension Center. At the top, there's a search bar and a 'Create SuccessFactors Extension' button. Below that, a navigation bar has 'SuccessFactors Extensions (9)' selected. A table lists 'Global Account in SAP BTP' and 'System Name' with all entries showing 'Integrated' status. A red box highlights the 'Integration Token' input field, which contains placeholder text: 'Enter a new integration token generated in SAP BTP cockpit'.

Global Account in SAP BTP	System Name	Integration Status
[REDACTED]	[REDACTED]	Integrated

4. In the cockpit, check the status of the registration process. To do so, navigate to your global account, and on the [Systems](#) page, check if the status of the SAP System has changed to [Registered](#).

If you are already on the [Systems](#) page, refresh the page to check if the status has changed.

#### ⓘ Note

You can register a system only once with the same name per global account.

#### ⓘ Note

You cannot migrate the registered SAP SuccessFactors systems between global accounts.

If you want to start using another global account, you will have to register your SAP SuccessFactors systems again.

## Related Information

[Registering and Deregistering Systems \[page 1997\]](#)

[Deregistering or Removing a System \[page 2019\]](#)

## 5.3.4 Register an SAP Customer Experience System

Register an SAP Customer Experience system to connect it with a global account in SAP BTP.

### Prerequisites

- See [Registering and Deregistering Systems \[page 1997\]](#).
- You are an administrator of the global account where you want to register your SAP Customer Experience system.
- To configure the integration on the SAP Customer Experience system side, you need to be an administrator of the SAP Customer Experience tenant.

### Context

The registration process is based on a registration token that is used for the pairing of the SAP Customer Experience system and the corresponding global account. You create the token in the global account, and then the tenant administrator of the respective SAP Customer Experience system uses the token to start the automated registration process on the SAP Customer Experience system side.

The registration process has the following states displayed in the cockpit:

- No status displayed in the *Status* column - the registration token for an SAP system has been created but the registration on the respective SAP solution system side has not been performed or completed.
- *Registered* - the registration token has been used and the automated registration process has been successfully completed. The system can be assigned to a formation on the *Formations* page in the cockpit.
- *Error while Registering* - the registration has failed. Remove the system and then add it to the *Systems* list and try to register it again
- *Deregistering* - a deregistration process has started in the SAP BTP cockpit. As a result, the connection between the SAP solution system and the global account in SAP BTP is removed. The system remains in the list and you can register it again later on.

Once a system is registered, you can deregister it only after removing it from all entitlement configurations and formations it takes part in.

- *Error while Deregistering* - the deregistration has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-CP-MP component.
- *Removing* - a system removal process has started in the SAP BTP cockpit. As a result, the link between the SAP solution and SAP BTP is destroyed and the system is removed from the list. To register the system again, first you must add it to the list anew, and then initiate the registration procedure.

Once a system is registered, you can only remove it if you first deregister it.

#### Note

You will not be able to remove a system if its status is one of the following:

- *Registered*  
You first need to deregister the system.

- *Deregistering*
- *Error while Deregistering*  
Try to deregister the system again.
- *Error while Removing* - the system removal has failed. Try to deregister the system again. If the problem persists, you have to report a case in the BC-CP-MP component.

#### ⓘ Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. In the cockpit, navigate to your global account, and then choose ► *System Landscape* ► *Systems* ▶.
2. On the *Systems* page, choose *Add System*.
  - a. Enter a name for the system you want to register.

#### ⓘ Note

Use only printable ASCII characters.

#### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, `<mysystem>-S/4HANA-cloud`. This helps you identify the system type when assigning systems to a formation.

- b. In the *Type* dropdown list, select the system type.

The following SAP Customer Experience systems are supported:

- SAP Commerce Cloud
- SAP Field Service Management
- SAP Cloud for Customer

- c. Choose *Add*.
- d. Choose *Get Token*.

The system generates the registration token.

- e. Copy the registration token and send it to the tenant administrator for the respective SAP Customer Experience system. You need it for configuring the integration on the extended SAP Customer Experience system side.

You can also get the registration token later, once the system appears in the list on the *Systems* page.

The integration token is valid for 10 minutes after it has been generated. When a token is not used within its validity period, it is no longer valid and cannot be used for registering a system. If the validity of the token expires before you use it to configure the integration on the SAP Customer Experience system side and complete the registration, you need to get a new token. You can then copy it and use it to complete the registration.

### ⓘ Note

A registration token can be used only once, for registering a single SAP Customer Experience system.

- f. Close the wizard.

The SAP Customer Experience system appears in the list of systems on the [Systems](#) page. Its [Status](#) field is empty because the registration process is not yet completed.

3. (Recommended) Follow the steps in [Including Systems in a Formation \[page 2027\]](#) before proceeding with the registration on the SAP Customer Experience side.
4. Configure the integration on the SAP Customer Experience system side. See [Extending SAP Customer Experience Products in the Kyma Environment \[page 2136\]](#).
5. Check the status of the registration process. To do so, in the cockpit navigate to your global account, and in the [Systems](#) panel, check if the status of the SAP Customer Experience system has changed to [Registered](#).

If you are already in the [Systems](#) panel, refresh the page to check if the status has changed.

## Results

Once you use the integration token to connect your SAP Customer Experience system, all of the exposed services and events are propagated to the Kyma runtime.

## Related Information

[Registering and Deregistering Systems \[page 1997\]](#)

[Deregistering or Removing a System \[page 2019\]](#)

## 5.3.5 Registering a Third-Party System

To connect a third-party systems, for example a Google system, with a global account in SAP BTP, you first need to add this system to the [Systems](#) page.

## Prerequisites

You are a global account administrator, or you are a system landscape administrator of the global account where you want to add your third-party system. See [Working with Role Collections \[page 2274\]](#).

## Context

### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

You add a third-party system to the list in the [System Landscape](#) [Systems](#) page. At this point you provide all the required details for this system: its type, provider, URL, and system ID. For third-party systems, this completes the registration process and you have your third-party system registered with SAP BTP. Even though the third-party system is registered directly, no status is displayed.

When you have this system added in the [Systems](#) page, you can select it and open its system details. There, you specify in the consumption bundles the APIs, and the events. A consumption bundle organizes a set of related APIs and events into a single group for consumption purposes and expresses information about how the APIs and events that it contains can be accessed. All APIs and events that are part of the same consumption bundle need to be accessible through the same set of credentials.

### ⓘ Note

You cannot migrate the added third-party systems between global accounts.

If you want to start using another global account, you will have to add your systems again.

### ⓘ Note

When adding a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. In the cockpit, navigate to your global account, and then choose [System Landscape](#) [Systems](#).
2. On the [Systems](#) page, choose [Add System](#).
3. In the [Add System](#) wizard:
  - a. Enter a name for the system you want to register.

### ⓘ Note

Use only printable ASCII characters.

### → Tip

We recommend that you indicate the type of the system when specifying the system name. For example, `<mysystem>-google-workspace`.

- b. In the [Type](#) dropdown list, select [Other System Type](#).
- c. In the [System Type Name](#) field, enter the type of your system. This is a user-defined type of the system you want to add. For example, [Workspace](#).

- d. In the *Provider* field, enter the provider of your system. For example, *Google*.
  - e. In the *URL* field, enter the URL of your system.
  - f. In the *System ID* field, enter the unique identifier of the added system that is used in its own domain.
  - g. Choose *Add*.
4. Find this system in the *Systems* list and open its system details. Choose *Add Consumption Bundle* and fill in the following properties:
- *Consumption Bundle Name*: add a meaningful name
  - *Description*:
  - *Credentials Type*: choose between OAuth and Basic
    - If you choose *OAuth*, fill in these properties:
      - *Client ID*
      - *Client Secret*
      - *URL*: this is the token URL, that is used for obtaining OAuth tokens through the client ID and the client secret.
    - If you choose *Basic*, fill in these properties:
      - *User*
      - *Password*
5. Choose *Add*. Open the consumption bundle from the list.
6. If you want to add APIs, choose *Add API* and fill in the following properties:
- *API Name*: add a meaningful name.
  - *Description*: this field is optional.
  - *URL*
  - *Specification*: this is the file containing the specification. The file format depends on the specification type. For OpenAPI, you can use YAML and JSON. For OData, you can use JSON and XML.
  - *Type*: this is the type of the specification. The API specification type can be OpenAPI or OData.
7. If you want to add events, choose *Add Event* and fill in the following properties:
- *Event Name*: add a meaningful name.
  - *Description*: this field is optional.
  - *Specification*: this is the file containing the specification. The file format can be JSON and YAML.
  - *Type*: this is the type of the specification. The event specification type can be only AsyncAPI.

## Results

The system has been added as a record to the list on the *Systems* page in the SAP BTP cockpit and you have added the respective APIs or events.

If you no longer need it, you can remove the system depending on its status. See [Deregistering or Removing a System \[page 2019\]](#).

## 5.3.6 Deregistering or Removing a System

When you no longer need the system to be paired with your global account, you can deregister or remove it depending on its status.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator of the global account where you want to deregister or remove your system. See [Working with Role Collections \[page 2274\]](#).
- You have added your system as a record to the list on the **Systems** page. See [Registering and Deregistering Systems \[page 1997\]](#).

### Context

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Deregistering an SAP or third-party system means that the connectivity between this system and the global account is disabled and extension scenarios cannot be established, while removing a system means that this system is no longer part of the system landscape list. You can deregister or remove a system from the **Actions** column, or from the **System Details** page that you access when selecting the system from the system landscape list.

The deregistration and the removal processes have the following states displayed in the cockpit:

- **Deregistering** - a deregistration process has started. As a result, the connectivity between the SAP or third-party system and SAP BTP is disabled and extension scenarios cannot be established. The system remains in the system landscape list and you can register it again later on.  
Once a system is registered, you can deregister it only after removing it from all entitlement configurations and formations it takes part in.
- **Error while Deregistering** - the deregistration has failed. Try to deregister the system again. If the problem persists, you have to report a case in one of the components mentioned below.
- **Removing** - a system removal process has started in the SAP BTP cockpit. As a result, the SAP or third-party system is deregistered, and then, it is removed from the system landscape list completely. To register the system again, first, you must add it to the list anew, and then, initiate the registration procedure.  
Once a system is registered, you can only remove it if you first deregister it.
- **Error while Removing** - the system removal has failed. Try to deregister the system again. If the problem persists, you have to report a case in one of the components mentioned below.

#### ⓘ Note

If the deregistration, or removal errors persist, report a case in one of the following components depending on the system type:

- SAP SuccessFactors - BC-NEO-EXT-SF
- SAP S/4HANA Cloud - BC-NEO-EXT-S4C
- All other system types - BC-CP-MP

## Procedure

1. In the cockpit, navigate to your global account, and then choose  [System Landscape](#)  [Systems](#).
2. On the [Systems](#) page, select the SAP or third-party system you want to deregister or remove.
  - To deregister a system, choose  [\(Deregister system\)](#) from the [Actions](#) column, or choose [Deregister System](#) from the [System Details](#) page that you access when selecting the system from the system landscape list.You can deregister only registered systems.

### Note

If your system is of type SAP S/4HANA Cloud, SAP SuccessFactors, or part of the SAP Customer Experience portfolio and you deregister this system, you can register back the same system's tenant and not a different one.

- To remove a system, choose  [\(Remove system\)](#) from the [Actions](#) column, or choose [Remove System](#) from the [System Details](#) page that you access when selecting the system from the system landscape list.
- Before removing a system from the list, first you have to check all configurations that this system is part of, such as entitlements and formations. Then, you have to deregister it and after that you will be able to remove it from the [Systems](#) page. If you try to remove a system before deregistering it, a dialog will appear and will ask you to deregister the system first.

### Note

You can only remove manually added systems. Depending on their discovery mode the systems that are added to the list either are [Manually added](#) (registered manually by following the procedure in the prerequisites), or [Auto-discovered](#) (added to the list automatically, based on information of the existing system landscape), or part of the [Subaccount/<my-subaccount>](#) (automatically added, based on the information of the SAP BTP subaccount).

You cannot remove [Auto-discovered](#) systems from the list.

## Results

The system has been deregistered or removed as a record from the list on the [Systems](#) page in the SAP BTP cockpit.

## 5.3.7 Merging SAP Systems

Merging the SAP systems allows you to reduce the duplicate systems in the list. You can merge auto-discovered SAP systems without assigned status in a target system that has already been registered.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator of the global account where you want to deregister or remove your SAP system. See [Working with Role Collections \[page 2274\]](#).
- You have registered the target SAP system. See [Registering and Deregistering Systems \[page 1997\]](#).

### Context

The logic of the SAP BTP cockpit enables it to discover SAP systems across your landscape and add them automatically on the [Systems](#) page. If you have already added these systems manually at a given point in time and you have registered them, the auto-discovered systems might create duplicate entries in the list. To avoid duplicates and streamline the list of systems, you can merge the auto-discovered SAP systems with your manually added systems. The logic automatically detects duplicates based on their URL and system type. Note that you can only merge auto-discovered SAP systems without assigned status in a target system that has already been registered. That is, you cannot merge registered auto-discovered systems.

### Procedure

1. In the cockpit, navigate to your global account, and then choose ► [System Landscape](#) ► [Systems](#) ▾.
2. On the [Systems](#) page, find the auto-discovered SAP system you want to merge.
3. Under the [Actions](#) column, choose the ➔ (*Merge system*) button.

Alternatively, you can choose the system from the list to open the [System Details](#) view, and then, choose the [Merge System](#) button on the top-right corner.

The [Merge System](#) dialog opens.

4. In the [Target System](#) dropdown menu, find and select the SAP system, in which you want to merge your system.
5. Choose [Merge](#).

### Results

The auto-discovered system has been merged in the target system. That is, all individual properties of the system were recorded in the target system and the common properties were preserved. At the end, the

duplicate (auto-discovered) system has been removed as a record from the list on the *Systems* page in the SAP BTP cockpit.

## 5.4 Extending SAP Solutions

The extension capabilities of SAP Business Technology Platform (SAP BTP) enables developers to implement loosely coupled extension applications securely, thus implementing additional workflows or modules on top of the existing SAP cloud solution they already have.

### Overview

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

With the automated configuration you have the following key benefits:

- A way of extending standard SAP solutions without disrupting their performance and core processes
- Frameworks that offer simplified, standardized, and unified extensibility and configuration for SAP solutions
- A central repository for solutions' APIs, events, credentials, and other data, thereby ensuring easy access to services while creating your extensions

### Process Flow

You can use the extension capabilities of SAP BTP to implement additional workflows or modules on top of your existing SAP solutions. You can extend one or more SAP solutions grouped together in a common business case.

To enable the integration, you need to:

1. Connect the corresponding SAP system with the global account.
  - For systems of type SAP S/4HANA Cloud, SAP Marketing Cloud, SAP SuccessFactors and SAP systems from the SAP Customer Experience portfolio, during the pairing process you create an registration token which is then used by the SAP system administrator to configure the registration on the SAP system side. See [Registering and Deregistering Systems \[page 1997\]](#).
  - For systems of type different than SAP S/4HANA Cloud, SAP Marketing Cloud, SAP SuccessFactors and SAP systems from the SAP Customer Experience portfolio, the SAP system is already registered when it's part of the *Systems* page. However, in the *Status* column, there is no status shown. See [Registering and Deregistering Systems \[page 1997\]](#).
  - For third-party systems, the system is already registered when it's part of the *Systems* page. However, in the *Status* column, there is no status shown. See [Registering a Third-Party System \[page 2016\]](#).

## Note

You cannot migrate the registered SAP systems between global accounts.

If you want to start using another global account, you will have to register your SAP systems again.

2. Optional: Group SAP systems so that they can be extended in a business scenario at one go. To do so, you create a formation containing one or more different systems assigned to a common subaccount. See [Including Systems in a Formation \[page 2027\]](#).
3. Make the SAP system accessible in the subaccounts in which you want to build your extension applications. For systems of type SAP S/4HANA Cloud, SAP Marketing Cloud, and SAP SuccessFactors, configure the entitlements and assign the corresponding quota and service plans to the subaccounts where the extension applications will reside. The service plans define the access to the corresponding SAP solution APIs. See:
  - [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#)
  - [Configure the Entitlements for the SAP SuccessFactors Extensibility Service \[page 2114\]](#)For the systems of type different than SAP S/4HANA Cloud, SAP Marketing Cloud, and SAP SuccessFactors, you can continue with developing your extension application.
4. For systems of type SAP S/4HANA Cloud, SAP Marketing Cloud, and SAP SuccessFactors, configure the communication flow for the extension application. To be able to consume the SAP S/4HANA Cloud, SAP Marketing Cloud, and SAP SuccessFactors APIs, you need to create a service instance for the corresponding SAP solution system. During the creation of the service instance, you configure the communication flow between the subaccount and the corresponding SAP solution. An HTTP destination which contains the binding properties for establishing the connection is automatically generated. After you have created the service instance, you have two options to configure the extension application's connectivity to the corresponding SAP solution:
  - Consume the HTTP destination
  - Bind the service instance to the extension application

## Related Information

[Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#)

[Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 2112\]](#)

[Extending SAP Customer Experience Products in the Kyma Environment \[page 2136\]](#)

[Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment \[page 2108\]](#)

## 5.4.1 Declaring System APIs and Events as Dependencies for Business Scenarios

### Context

For certain formation types, systems of type *Other System Type*, *SAP BTP Application*, or *SAP Integration Suite* that are part of the *System Landscape* page may need to describe their runtime dependencies so other systems can comply with these dependencies to be able to interact with these systems in a common scenario.

The integration dependency includes a list of requirements, which point out which API and event resources are involved. Each requirement describes one aspect and can be used to express alternatives using OR conditions for achieving the same outcome.

To add the necessary API and event resources to an integration dependency, systems use the Open Resource Discovery standard. The Open Resource Discovery standard is about adding missing or common high-level information and standardizing the discovery aspects. See [Why ORD](#).

Every integration dependency has the following properties:

- Name: a human-readable name that does not exceed 255 characters. For example, [Integration Dependency to realize Customer Order data product](#).
- Description: a plain text that does not exceed 255 characters. For example, [SAP S/4HANA Cloud, our next generation cloud ERP suite designed for in-memory computing](#).
- Version: the version is assigned automatically when adding a new integration dependency.

#### ⓘ Note

If you add a new integration dependency using the same name as an existing integration dependency, the version of the new integration dependency will be automatically increased. Then, in the given system, there will be two integration dependencies with the same name and different versions.

Depending on your use case, there are different templates that help you set up the necessary integration dependency:

- Blank template  
You write integration dependencies from scratch including the aspects that contain a set of APIs and/or events of a system that define the dependencies other systems need to comply with to be able to be part of an integration scenario together.
- Simplified Business Eventing template  
Integration dependencies are exposed as available event subscriptions in SAP Cloud Application Event Hub. This template helps you write simplified integration dependencies that identify the types of events a subscribing application wants to consume. Here's an example. Let's say SAP SuccessFactors publishes a set of events. You want to receive a subset of those events. You can use the [Simplified Business Eventing Template](#) to identify the event types that SAP Cloud Application Event Hub makes available to your subscribing system.

## 5.4.1.1 Add an Integration Dependency Using a Blank Template

### Prerequisites

You have a system of type *Other System Type* or *SAP BTP Application* listed in the [Systems](#) page of your global account in SAP BTP. See [Registering a Third-Party System \[page 2016\]](#).

### Context

Integration dependencies contain aspects. An aspect contains a set of APIs and/or events of a system that define the dependencies other systems need to comply with to be able to be part of an integration scenario together.

### Procedure

1. In the cockpit, navigate to your global account, and then choose [System Landscape](#) [Systems](#).
2. On the [Systems](#) page, select a system of type *Other System Type* to which you want to add an integration dependency and open its system details.
3. In the [Integration Dependencies](#) tab, choose [Add](#).
4. Select [Blank Template](#) to write an integration dependency from scratch.
5. Define a name and a description for the new integration dependency. Choose [Next Step](#).

#### Note

The version is assigned automatically when adding a new integration dependency. If you add a new integration dependency using the same name as an existing integration dependency, the version of the new integration dependency will be automatically increased.

6. Define a name and a description for the first aspect and add more if necessary. Choose [Next Step](#).
7. Add at least one API or event to each aspect and choose [Review](#).
8. Check if all the aspects are configured properly and choose [Add](#).

#### Note

Once you add an integration dependency to a system, you can no longer edit it and add or delete aspects, APIs or events. You can only delete an integration dependency.

## 5.4.1.2 Add an Integration Dependency Using a Simplified Business Eventing Template

### Prerequisites

- An SAP Cloud system is registered in the [Systems](#) page in SAP BTP cockpit and has a catalog of events that it plans to publish. Refer to the product documentation for the publishing system for more details. For example, an SAP SuccessFactors system can be configured to publish events. See [SAP Cloud Application Event Hub Service Guide: Integration Use Cases](#).  
The SAP system you plan to use as the subscriber is registered in the [Systems](#) page. If you want to use SAP Integration Suite integration flows to consume events from SAP Cloud Application Event Hub, then the tenant with the integration flow must be registered in the [Systems](#) page. See [Integration Example using SAP Integration Suite](#).
- Your subscribing system is listed in the [Systems](#) page of your global account in SAP BTP.  
For example, you have a system of type [SAP Integration Suite](#) listed in the [Systems](#) page. It appears as auto-discovered.

### Context

You can subscribe to events with extension applications. For example, you use SAP SuccessFactors to publish events and an Integration Suite tenant to consume them. The SAP SuccessFactors system is publishing these events and you add the event types that you want this system to receive using the [Simplified Business Eventing Template](#) of the integration dependencies. Integration dependencies are exposed as available event subscriptions in SAP Cloud Application Event Hub. The [Simplified Business Eventing Template](#) helps you write simplified integration dependencies specifying only event-specific information.

### Procedure

1. In the cockpit, navigate to your global account, and then choose [System Landscape](#) [Systems](#).
2. On the [Systems](#) page, select a system of type [Integration Suite](#) to which you want to add an integration dependency and open its system details.
3. In the [Integration Dependencies](#) tab, choose [Add](#).
4. Select [Simplified Business Eventing Template](#) to identify the system that is publishing events and add the event types that you want to receive. Integration dependencies are exposed as available event subscriptions in SAP Cloud Application Event Hub.
5. Define a name and a description for the new integration dependency.

### ⓘ Note

The version is assigned automatically when adding a new integration dependency. If you add a new integration dependency using the same name as an existing integration dependency, the version of the new integration dependency will be automatically increased.

6. In the *Publishing System Namespace* field, provide the system namespace of the system that is publishing events. You can find the system namespace in the *System Details* of the system.

The system namespace has to follow these rules:

- Use only lowercase ASCII letters (a-z) and digits (0-9)
- Include at least one character
- Use the dot (.) only to separate fragments
- Include only two fragments separated by a dot  
Example: sap.foo

7. In the *Event Type* field, list the event types published by the system that you want to make available to your subscribing system.

You can write or paste the event types separated by comma, semicolon or new row. Preferably, each event type should be on a new row. You have a counter giving the number of event types that are already filled in.

8. Choose *Next Step*.
9. Check if all the event types are configured properly and choose *Add*.

### ⓘ Note

Once you add an integration dependency to a system, you can no longer edit it and add or delete event types. You can only delete the integration dependency itself.

## 5.4.2 Including Systems in a Formation

You can include various SAP systems into a formation and thus combine diverse SAP solutions into an extended business scenario.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- The SAP solution systems that you want to include to a formation must be added in the *Systems* page. See [Registering and Deregistering Systems \[page 1997\]](#).

## Context

### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

A formation is a logical grouping of SAP systems that can be extended in a business scenario. Formations allow you to combine SAP solution systems and a subaccount in SAP BTP to simplify the connectivity setup and to provide a unified view of all components required for the implementation of your extension scenario. To create a fully functional formation, you can use a two-step wizard. At the first step, you specify a custom formation name and assign a subaccount to it. At the second step, you can include an SAP solution system in the formation. You do this configuration once and you can change it anytime.

### ⓘ Note

You can assign a subaccount to the formation during the formation creation only. While you can include or exclude systems in the formation anytime, you cannot unassign or reassign subaccounts later on. Instead, you must recreate the formation.

Extension business cases often involve extending several SAP solutions at a time. For example, for a given business case you might need to extend the functionality or the UI as follows:

- An SAP SuccessFactors system, and an SAP S/4HANA Cloud system. First, you need to configure the connectivity of each of these systems to Cloud Foundry, Kyma, or both environments. Extension applications of both solutions are part of the same business need.
- An SAP Commerce Cloud system, and an SAP S/4HANA Cloud system. Again, you first configure the connectivity of each of these systems to the Kyma environment.
- A single system of the supported SAP solutions.
- SAP systems that expose event data, which can be shared and exchanged with the systems included in the formation.

When creating a formation in the SAP BTP cockpit, you include the systems of the different SAP solutions you want to extend. If your business case features more than one SAP solution system, you can use the corresponding button to include additional systems in the formation. You can start the dialog as many times and add systems to your formation as you want.

A formation can have the following status values:

- **No status**

The formation has been added as a record to the list and all registered systems are included successfully.

- **Action Required**

The formation has been created but you cannot use it productively yet.

Based on the formation type and the subaccount that you specified, it might require an SAP BTP Kyma environment instance or an SAP Business Application Studio subscription. Although the SAP BTP cockpit allows you to create such a formation, to enable and make use of it, you must also create the respective instance or subscription.

- **Synchronizing**

Systems that are included in the formation are currently synchronizing in the background.

- **Error**

An error occurred while some of the systems that are included in the formation were synchronizing in the background. In this case, report an incident in the BC-CP-MP component.

When you include systems in a formation, these systems are synchronized in the background. If an error occurs, you can resynchronize these systems to restart the synchronization process. To do that, choose [Resynchronize](#) for the particular formation. You will have this action available only for formations that are in status [Error](#).

If you want to reconfigure the systems in a formation from scratch without excluding them, you have to choose [Reset and Resynchronize](#). You will have this action available only for formations that have no status in SAP BTP cockpit and their formation type is Side-by-Side Extensibility with Kyma.

### Note

When registering a system or creating a formation, the data you provide in the given input fields is not encrypted with your customer managed key. The data you enter is only encrypted at rest.

## Procedure

1. Open the SAP BTP cockpit and navigate to your global account.
2. Choose  [System Landscape](#)  [Formations](#).
3. Choose [Create Formation](#).

The [Create Formation](#) dialog opens. There, in a wizard, you can create a formation and include one or more SAP systems to it.

4. On the [General Information](#) step of the wizard, enter the following information:
  - a. Enter a unique formation name.

The formation name can contain lowercase or uppercase Latin letters, numbers, hyphens, spaces, or underscores only. The name must start and end with an alphanumeric character.

- b. Specify a type for the formation.

The formation type defines the use case. Therefore, depending on the use case, you can specify one of the following formation types:

Formation Type	Description
<a href="#">Side-by-Side Extensibility with Kyma</a>	Formations of type <a href="#">Side-by-Side extensibility with Kyma</a> enable business scenarios that involve extending the functionality of several SAP systems at a time with SAP BTP Kyma environment instance.  See <a href="#">Enabling Side-by-Side Extensibility with Kyma [page 2032]</a> .

Formation Type	Description
<i>Developing with SAP Business Application Studio</i>	<p>Formations of type <i>Developing with SAP Business Application Studio</i> enable connectivity between given SAP systems of type <i>SAP S/4HANA Cloud</i> from the <i>System Landscape</i> page of SAP BTP cockpit and the SAP Business Application Studio, you must create a formation of the corresponding type and include the SAP S/4HANA Cloud systems in it.</p> <p>See <a href="#">Enabling System Landscape for SAP Business Application Studio [page 2033]</a>.</p>
<i>Integration with SAP Build</i>	<p>Formations of type <i>Integration with SAP Build</i> enable connectivity between given SAP systems of type <i>SAP S/4HANA Cloud</i> from the <i>System Landscape</i> page of SAP BTP cockpit and SAP Build. You must create a formation of the corresponding type and include the SAP S/4HANA Cloud systems in it.</p> <p>See <a href="#">Enabling System Landscape for SAP Build [page 2035]</a>.</p>
<i>Integration with SAP Integration Suite</i>	<p>Formations of type <i>Integration with SAP Integration Suite</i> enable the communication flow between SAP S/4HANA Cloud systems and SAP Integration Suite.</p> <p>See <a href="#">Enabling System Landscape for SAP Integration Suite [page 2037]</a>.</p>
<i>Eventing Between SAP Cloud Systems</i>	<p>Formations of type <i>Eventing Between SAP Cloud Systems</i> enable end-to-end communication flow of events between the SAP systems that are included in the given formation. As the name of the formation suggests, the systems transmit information about events using SAP Cloud Application Event Hub. To be able to exchange such events information, the SAP systems must be also visible and configurable in the SAP Cloud Application Event Hub user interface. The formation makes the systems visible in SAP Cloud Application Event Hub, however, it requires a subscription to SAP Cloud Application Event Hub in the corresponding SAP BTP subaccount first.</p> <p>See <a href="#">Enabling Events Exchange Between SAP Cloud Systems [page 2038]</a>.</p>

Formation Type	Description
<i>Integration with SAP Ariba Buying</i>	Formations of type <i>Integration with SAP Ariba Buying</i> set up SAP Ariba Buying with SAP S/4HANA Cloud on SAP BTP.  See <a href="#">Enabling SAP Ariba Buying [page 2040]</a> .
<i>Integration with SAP Start</i>	Formations of type <i>Integration with SAP Start</i> set up SAP Build Work Zone, standard edition with SAP S/4HANA Cloud on SAP BTP.  See <a href="#">Enabling SAP Start [page 2042]</a> .
<i>Data Ingestion for Industry Cloud Solutions</i>	Formations of type <i>Data Ingestion for Industry Cloud Solutions</i> set systems that support data ingestion and are part of the industry cloud solutions published by SAP and SAP BTP.  See <a href="#">Enabling Data Ingestion for Industry Cloud Solutions [page 2044]</a> .

Depending on the type, a list of systems that can be included in this formation is loaded at the following step of the wizard.

- c. Specify a subaccount that you want to associate with the formation.

All systems that you include in the formation will be available and can be managed in the respective application or runtime in this subaccount.

#### ① Note

Based on the formation type and the subaccount that you specified in the newly created formation, it might require an SAP BTP Kyma environment instance. Although the SAP BTP cockpit allows you to create such a formation, to enable and make use of it, you must also create an SAP BTP Kyma environment instance for the corresponding SAP BTP subaccount.

5. Choose *Next Step*.
6. On the *Include Systems* step, select one or more systems that you want to include in the newly created formation, and then, choose *Next Step*.  
  
The wizard prefilters the systems that were added to the *Systems* page and are valid for the formation type that you specified at the previous step.
7. On the *Review* step, double check your entries before you create the formation.  
  
If you want to make any changes, either you can edit the corresponding section directly, or use the *Previous* button.
8. Choose *Create*.
9. Optionally, you can include additional systems to the newly created formation, by choosing *Include System*.  
  
On the systems list that opens, select a system, and then, choose *Include*.

## Results

For systems of type SAP Commerce Cloud, SAP Cloud for Customer, and SAP Field Service Management, the access to the corresponding solution's APIs has been enabled. After you have created a formation, you can edit it and change the included systems. The status of each system depends on whether you have registered that system in the global account.

### ⓘ Note

When you delete a formation, several activities are performed at one go. First, the systems are excluded from the formation. Then, the subaccount is unassigned. Finally, the formation is deleted from the [Formations](#) page completely.

To restore a deleted formation, first you must create it anew, and then, include all of its systems again, one by one.

### 5.4.2.1 Enabling Side-by-Side Extensibility with Kyma

## Use

### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

You want to extend the functionality on top of your existing SAP solution because your specific business needs require it. You also want to use the SAP BTP Kyma environment. You can build an extension application that contains your business logic and UI changes and deploy it in SAP BTP Kyma runtime. For the users of your SAP solution, this change will be seamless, it will look as if it's part of the original functionality. At the same time, you can use the extension capabilities of SAP BTP to implement the additional workflows or modules. This is called side-by-side extensibility with Kyma.

## Procedure

To enable the side-by-side extensibility with Kyma for your SAP solution, you have to:

1. Create an SAP BTP, Kyma environment instance in your subaccount in SAP BTP.
2. Add the Application Connector module on your Kyma runtime.
3. Connect the corresponding SAP system with the global account in SAP BTP. Use the automated configurations for the following SAP solutions:
  - [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#)
  - [Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment \[page 2108\]](#)

- [Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 2112\]](#)
  - [Extending SAP Customer Experience Products in the Kyma Environment \[page 2136\]](#)
4. Create a formation of type *Side-by-Side Extensibility with Kyma* where you can include one or more SAP systems depending on your use case.

## 5.4.2.2 Enabling System Landscape for SAP Business Application Studio

### Use

#### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

A registered SAP S/4HANA Cloud system in the SAP BTP cockpit can expose consumption bundles that contain APIs and events. You can easily discover and consume the APIs exposed by the SAP S/4HANA Cloud systems in your system landscape when you develop and extend applications on SAP BTP, Cloud Foundry environment, using SAP Business Application Studio. To do this, first you need to enable connectivity between your system landscape in SAP BTP cockpit and SAP Business Application Studio. The integration requires performing several configuration steps starting with configuration on a global account level, and then, configuration on a subaccount level.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have an SAP Business Application Studio subscription. See [Subscribe to SAP Business Application Studio](#).

### Procedure

The following procedure outlines the steps you need to perform to consume the APIs of registered SAP S/4HANA Cloud systems within the SAP Business Application Studio.

1. In the   page of SAP BTP cockpit, add and then register an SAP system of type *SAP S/4HANA Cloud*.

To expose information about its APIs and events and show this information on the *Systems* page, an SAP system of type *SAP S/4HANA Cloud* must be registered in the SAP BTP cockpit. Only when registered, the

system communicates information about its APIs and other technical details across the landscape. See [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#).

2. In the page, create a formation of type *Developing with SAP Business Application Studio* and include the SAP Business Application Studio system and the SAP S/4HANA Cloud systems you want to expose in SAP Business Application Studio.

#### Note

In the formation type *Developing with SAP Business Application Studio*, you can include only SAP systems of type *SAP Business Application Studio* and *SAP S/4HANA Cloud*.

To enable connectivity between given SAP systems of type *SAP S/4HANA Cloud* from the *Systems* page of SAP BTP cockpit and the SAP Business Application Studio, you must create a formation of the corresponding type and include the SAP S/4HANA Cloud systems in it. See [Including Systems in a Formation \[page 2027\]](#).

3. Create a destination and make sure it has the corresponding system and consumption bundle properties.
  - Create a destination manually.  
You can create the destination manually via the SAP Destination service page in SAP BTP cockpit by specifying one of the following additional properties variants in it.

Additional Properties (variant 1)	Additional Properties (variant 2)	Description
x-correlation-id	x-correlation-id	The property identifies the consumption bundle and its APIs that are exposed by the SAP S/4HANA Cloud system. The destination also provides the required credentials to consume the bundle and use it for further development in the SAP Business Application Studio.
x-system-id	x-system-name	Properties that uniquely identify the local tenant of the registered SAP S/4HANA Cloud system.
x-system-type	x-system-base-url	

All of the required properties of a given SAP S/4HANA Cloud system are accessible in the corresponding *System Details* section on the *Systems* page. See [Create a Destination \[page 891\]](#).

- Create a destination automatically.  
Alternatively, the destination can be created automatically. You can do this, by creating a service instance of the SAP S/4HANA Cloud Extensibility service after you register an SAP S/4HANA Cloud system. See [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#) and [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 2050\]](#).

## Next Steps

Develop applications on SAP BTP, Cloud Foundry environment using SAP Business Application Studio and the system landscape.

When you enable the connectivity, the SAP S/4HANA Cloud systems and their APIs are accessible in the SAP Business Application Studio. Next you can use the APIs to develop new or extend the existing functionality with the help of SAP Business Application Studio. See [Unified Customer Landscape Service Provider](#).

### 5.4.2.3 Enabling System Landscape for SAP Build

#### Use

##### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

A registered SAP S/4HANA Cloud system in the SAP BTP cockpit can expose consumption bundles that contain APIs and events. You can easily discover and consume the APIs exposed by the SAP S/4HANA Cloud systems in your system landscape when you develop and extend applications on SAP BTP, Cloud Foundry environment, using SAP Build. To do this, first you need to enable connectivity between your system landscape in SAP BTP cockpit and SAP Build. The integration requires performing several configuration steps starting with configuration on a global account level, and then, configuration on a subaccount level.

#### Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have an SAP Build subscription.

#### Procedure

The following procedure outlines the steps you need to perform to consume the APIs of registered SAP S/4HANA Cloud systems within SAP Build.

1. Add and then register an SAP system of type [SAP S/4HANA Cloud](#) in the  [System Landscape](#)  [Systems](#)  page of SAP BTP cockpit.  
To expose information about its APIs and events and show this information on the [Systems](#) page, an SAP system of type [SAP S/4HANA Cloud](#) must be registered in the SAP BTP cockpit. Only when registered, the system communicates information about its APIs and other technical details across the landscape. See [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#).
2. In the [Formations](#) page, create a formation of type [Integration with SAP Build](#) and include the SAP Build system and the SAP S/4HANA Cloud systems you want to expose in SAP Build.

## ⓘ Note

In the formation type *Integration with SAP Build*, you can include only SAP systems of type *SAP Build* and *SAP S/4HANA Cloud*.

To enable connectivity between given SAP systems of type *SAP S/4HANA Cloud* from the *Systems* page of SAP BTP cockpit and SAP Build, you must create a formation of the corresponding type and include the SAP S/4HANA Cloud systems in it. See [Including Systems in a Formation \[page 2027\]](#).

3. Create a destination and make sure it has the corresponding system and consumption bundle properties.

- Create a destination manually.

You can create the destination manually via the SAP Destination service page in SAP BTP cockpit by specifying one of the following additional properties variants in it.

Additional Properties (variant 1)	Additional Properties (variant 2)	Description
x-correlation-id	x-correlation-id	The property identifies the consumption bundle and its APIs that are exposed by the SAP S/4HANA Cloud system. The destination also provides the required credentials to consume the bundle and use it for further development in SAP Build.
x-system-id	x-system-name	Properties that uniquely identify the local tenant of the registered SAP S/4HANA Cloud system.
x-system-type	x-system-base-url	

All of the required properties of a given SAP S/4HANA Cloud system are accessible in the corresponding *System Details* section on the *Systems* page. See [Create a Destination \[page 891\]](#).

- Create a destination automatically.

Alternatively, the destination can be created automatically. You can do this, by creating a service instance of the SAP S/4HANA Cloud Extensibility service after you register an SAP S/4HANA Cloud system. See [Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 2046\]](#) and [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 2050\]](#).

## Next Steps

Develop applications on SAP BTP, Cloud Foundry environment using SAP Build and the system landscape.

When you enable the connectivity, the SAP S/4HANA Cloud systems and their APIs are accessible in SAP Build. Next you can use the APIs to develop new or extend the existing functionality with the help of SAP Build.

## Related Information

[SAP Build Process Automation: Using SAP Systems](#)

## 5.4.2.4 Enabling System Landscape for SAP Integration Suite

### Use

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

A registered SAP S/4HANA Cloud system in the SAP BTP cockpit can expose consumption bundles that contain APIs and events. You can easily discover and consume the APIs exposed by the SAP S/4HANA Cloud systems in your system landscape when you develop and extend applications on SAP BTP, Cloud Foundry environment. To do this, first you need to enable connectivity between your system landscape in SAP BTP cockpit and SAP Integration Suite. The integration requires performing several configuration steps starting with configuration on a global account level, and then, configuration on a subaccount level.

### Rules

When creating *Integration with SAP Integration Suite* formations, keep in mind the following rule:

- At most one system of type *SAP Integration Suite* can be included in one *Integration with SAP Integration Suite* formation.

### Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have a subscription to SAP Integration Suite. See [Subscribing and Configuring Initial Access to SAP Integration Suite](#).

#### ⓘ Note

Every time you create a subscription to a new SAP Integration Suite tenant, the system type for the tenant is automatically defined as SAP Integration Suite.

### Procedure

The following procedure outlines the steps you need to perform to consume the APIs of registered SAP S/4HANA Cloud systems within SAP Integration Suite.

1. Add and then register an SAP system of type [SAP S/4HANA Cloud](#) in the [System Landscape](#) page of SAP BTP cockpit.  
To expose information about its APIs and events and show this information on the [Systems](#) page, an SAP system of type [SAP S/4HANA Cloud](#) must be registered in the SAP BTP cockpit. Only when registered, the system communicates information about its APIs and other technical details across the landscape. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).
2. In the [System Landscape](#) [Formations](#) page, create a formation of type [Integration with SAP Integration Suite](#) and include the SAP S/4HANA Cloud systems you want to expose to SAP Integration Suite. See [Including Systems in a Formation \[page 2027\]](#).

**Note**

In the formation type [Integration with SAP Integration Suite](#), you can include only SAP systems of type [SAP S/4HANA Cloud](#).

### 5.4.2.5 Enabling Events Exchange Between SAP Cloud Systems

#### Use

**Note**

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Systems that are included in [Eventing Between SAP Cloud Systems](#) formations on the [System Landscape](#) page of the SAP BTP cockpit can publish and consume events. The process of publishing and consuming events is in fact an exchange of event information across the customer system landscape and is driven by the system formations on the one hand side and the SAP Cloud Application Event Hub on the other.

**Note**

By creating a formation of type [Eventing Between SAP Cloud Systems](#), you understand that there may be event distribution charges incurred if you exceed the free quota limit when using SAP Cloud Application Event Hub. To learn more about usage and consumption costs, see [Usage and Consumption Costs](#).

#### Rules

When creating [Eventing Between SAP Cloud Systems](#) formations, keep in mind the following rule:

- Every system can be included in at most one [Eventing Between SAP Cloud Systems](#) formation.

## Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have subscribed to the SAP Cloud Application Event Hub application. See [Subscribing to SAP Cloud Application Event Hub Application](#).

## Procedure

The following procedure outlines the steps you need to perform to enable the exchange of events across the systems within the system landscape.

1. In the SAP BTP cockpit, in the [System Landscape](#) [Systems](#) page of the SAP BTP cockpit, browse the already added systems in your customer system landscape or manually add and register any missing systems.

The customer landscape features systems that are added to the list in one of the following ways:

- ***Auto-discovered***  
An auto-discovered system is a system (associated with the given global account) that has been discovered and added automatically to the list based on information of the existing system landscape. Any SAP system of the supported system types that is associated with the same customer ID, with which your global account in SAP BTP is associated, will be added automatically in the system landscape list.
- ***Subaccount/<my-subaccount>***  
A subscription in SAP BTP cockpit associated with a given subaccount. The subscription has been discovered and added automatically through the subaccount. Check the value of the *Discovery* column to see the subaccount where your system is subscribed.
- ***Manually added***  
Specifies that the system has been added to the list manually by the global account administrator, using the [Add System](#) button and completing the wizard. The system has been associated with the global account in SAP BTP.

### Note

If a given SAP system is missing on the [Systems](#) page, it may be associated with a different customer ID on the SAP BTP global account you are working in. In this case, you need to add the system manually, and then, register it.

See [Registering and Deregistering Systems \[page 1997\]](#).

2. Create a formation of type [Eventing Between SAP Cloud Systems](#) and include the relevant systems in it.
  1. Add any name that helps you identify your formation.
  2. In the *Formation Type* dropdown menu, select [Eventing Between SAP Cloud Systems](#).
  3. Select the systems that you want to include in the formation. One of these systems must be of type [SAP Cloud Application Event Hub](#).

### Note

Systems can only be added to one formation of type [Eventing Between SAP Cloud Systems](#) in a global account.

Also, a formation of type *Eventing Between SAP Cloud Systems* can contain only one system of type *SAP Cloud Application Event Hub*.

4. Review your selections and create the formation.

## Next Steps

When the formation is created, navigate to the SAP Cloud Application Event Hub application and enable the event subscriptions. See [Enabling SAP Event Subscriptions](#).

## Remove a System from the Formation

If you want to remove an SAP system from a formation of type *Eventing Between SAP Cloud Systems*, make sure that all active event subscriptions are disabled first in SAP Cloud Application Event Hub. You can check your subscriptions in the SAP Cloud Application Event Hub application on the [Subscriptions](#) page. See [Disabling SAP Event Subscriptions](#).

### 5.4.2.6 Enabling SAP Ariba Buying

#### Use

##### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

SAP Ariba Buying offers procurement users a unique buying experience through a guided process. SAP Ariba Buying provides buyers an easy process to shop for items. Buyers can search for, view and purchase items that are available in the connected shopping sites, or request items from suppliers. SAP Ariba Buying also helps buyers to track their purchase requests.

To set up SAP Ariba Buying with SAP S/4HANA Cloud on SAP BTP you need to have all the necessary systems included in a formation of type *Integration with SAP Ariba Buying* in the  [System Landscape](#)  [Systems](#)  page of the SAP BTP cockpit.

## Rules

When creating *Integration with SAP Ariba Buying* formations, keep in mind the following rules:

- SAP S/4HANA Cloud systems can be included in at most one *Integration with SAP Ariba Buying* formation.
- Only SAP S/4HANA Cloud systems that are registered with *All Communication Scenarios* or *Integration with SAP Ariba Buying* communication scenario group can be part of a *Integration with SAP Ariba Buying* formation. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

## Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have an Identity Authentication tenant. See [Identity Authentication: Initial Setup](#).
- (Optional: if you are using the *Set Up SAP Ariba Buying* booster in the SAP BTP cockpit, this step will be automatically done by the booster.) Check that you have a system of type *SAP Ariba Buying* in the *Systems* page. This system is added through a subscription in SAP BTP cockpit associated with a given subaccount. The subscription has been discovered and added automatically through the subaccount. To have a system of type *SAP Ariba Buying* in the *Systems* page, you need to:
  - You have subscribed to SAP Ariba Buying. See [Get a Subscription to SAP Ariba Buying](#).
  - You have subscribed to the services required for SAP Ariba Buying. See [Subscribe to Services Required for SAP Ariba Buying](#).
- Check that you have a system of type *SAP S/4HANA Cloud* in the *Systems* page. This system is auto-discovered because it is associated with your global account and has been discovered and added automatically to the list based on information of the existing system landscape.

### ⓘ Note

If you don't find your system of type *SAP S/4HANA Cloud* in the *Systems* page, add this system manually. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

- Register the SAP S/4HANA Cloud system in the *Systems* page.  
When you register an SAP S/4HANA Cloud system, use the *Integration with SAP Ariba Buying* communication scenario group when you get the registration token for the SAP S/4HANA Cloud system. This allows the automatic enablement of the communication scenarios *SAP\_COM\_0545* and *SAP\_COM\_0A00* after the corresponding system is added to the formation of type *Integration with SAP Ariba Buying*. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

## Procedure

If you have SAP Ariba Buying set up in your subaccount in SAP BTP, follow these steps to configure the integration between SAP Ariba Buying and SAP S/4HANA Cloud using a formation of type *Integration with SAP Ariba Buying*.

If you want to have the end-to-end setup automatically, including creating the formation, use the [Set Up SAP Ariba Buying](#) booster in the SAP BTP cockpit to guide you through the steps to configure SAP Ariba Buying in your subaccount and integrate it with SAP S/4HANA Cloud.

1. In the [Systems](#) page, find the systems of type [SAP Ariba Buying](#) and [SAP S/4HANA Cloud](#) that you want to include in a formation.
2. In the  [System Landscape](#)  [Formations](#)  page, create a formation of type [Integration with SAP Ariba Buying](#) and include the relevant systems in it.
  1. Add any name that helps you identify your formation.
  2. In the [Formation Type](#) dropdown menu, select [Integration with SAP Ariba Buying](#).
  3. Select the [SAP Ariba Buying](#) and [SAP S/4HANA Cloud](#) systems that you want to include in the formation.
  4. Review your selections and create the formation.

## Next Steps

When the formation is created, you have SAP Ariba Buying set up with SAP S/4HANA Cloud on SAP BTP.

### 5.4.2.7 Enabling SAP Start

#### Use

##### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

SAP Build Work Zone, standard edition is a predefined central entry point for accessing SAP cloud business solutions that are integrated with it, such as SAP S/4HANA Cloud

To set up SAP Build Work Zone, standard edition with SAP S/4HANA Cloud on SAP BTP you need to have all the necessary systems included in a formation of type [Integration with SAP Start](#) in the  [System Landscape](#)  [Systems](#)  page of the SAP BTP cockpit.

#### Rules

When creating [Integration with SAP Start](#) formations, keep in mind the following rules:

- SAP Build Work Zone systems can be included in at most one [Integration with SAP Start](#) formation.
- At most one SAP Build Work Zone system can be included in an [Integration with SAP Start](#) formation.

- Only SAP S/4HANA Cloud systems that are registered with *All Communication Scenarios* or *Integration with SAP Start* communication scenario group can be part of a *Integration with SAP Start* formation. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

## Prerequisites

- You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).
- You have an Identity Authentication tenant. See [Identity Authentication: Initial Setup](#).
- (Optional: if you are using the *Integration with SAP Start* booster in the SAP BTP cockpit, this step will be automatically done by the booster.) Check that you have a system of type *SAP Build Work Zone* in the *Systems* page. This system is added through a subscription in SAP BTP cockpit associated with a given subaccount. The subscription has been discovered and added automatically through the subaccount.
- Check that you have a system of type *SAP S/4HANA Cloud* in the *Systems* list in the *Systems* page. This system is auto-discovered because it is associated with your global account and has been discovered and added automatically to the list based on information of the existing system landscape.

### Note

If you don't find your system of type *SAP S/4HANA Cloud* in the *Systems* list, add this system manually. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

- Register the SAP S/4HANA Cloud system in the *Systems* list in the *System Landscape* page. When you register an SAP S/4HANA Cloud system, use the *Integration with SAP Start* communication scenario group when you get the registration token for the SAP S/4HANA Cloud system. This allows the automatic enablement of the communication scenario *SAP\_COM\_0647* after the corresponding system is added to the formation of type *Integration with SAP Start*. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

## Procedure

If you have SAP Build Work Zone set up in your subaccount in SAP BTP, follow these steps to configure the integration between SAP Build Work Zone and SAP S/4HANA Cloud using a formation of type *Integration with SAP Start*.

If you want to have the end-to-end setup automatically, including creating the formation, use the *Integration with SAP Start* booster in the SAP BTP cockpit to guide you through the steps to configure SAP Build Work Zone in your subaccount and integrate it with SAP S/4HANA Cloud.

- In the SAP BTP cockpit, in the *Systems* page, find the systems of type *SAP Build Work Zone* and *SAP S/4HANA Cloud* that you want to include in a formation.
- In the  *System Landscape*  page, create a formation of type *Integration with SAP Start* and include the relevant systems in it.
  - Add any name that helps you identify your formation.
  - In the *Formation Type* dropdown menu, select *Integration with SAP Start*.

3. Select the [SAP Build Work Zone](#) and [SAP S/4HANA Cloud](#) systems that you want to include in the formation.
4. Review your selections and create the formation.

## Next Steps

When the formation is created, you have SAP Build Work Zone set up with SAP S/4HANA Cloud on SAP BTP. Then, set up SAP Task Center. See [Post-Setup Tasks](#).

### 5.4.2.8 Enabling Data Ingestion for Industry Cloud Solutions

#### Use

##### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

SAP's industry cloud provides specialized industry-focused solutions to help you optimize, extend, and transform your core business processes. Developed by SAP and SAP partners and based on SAP Business Technology Platform (SAP BTP), industry cloud solutions integrate with SAP S/4HANA Cloud and SAP Business Network. They help you take advantage of the latest industry-relevant innovations and extend existing capabilities in an efficient way. For a list of all industry cloud solutions published by SAP, see [SAP Store](#).

Data ingestion for industry cloud solutions allows supported industry cloud solutions to consume data from a shared data foundation instead of integrating with the source systems individually. This helps to simplify solution implementation, increase operational visibility, and improve data quality. For a list of all industry cloud solutions published by SAP that support data ingestion, see [Data Integration Overview](#).

To set up on SAP BTP systems that support data ingestion and are part of the industry cloud solutions published by SAP, you need to include them in a formation of type [Data Ingestion for Industry Cloud Solutions](#). When you create a new formation of type [Data Ingestion for Industry Cloud Solutions](#), a system of type [Data Ingestion for Industry Cloud Solutions](#) is automatically added to the  [System Landscape](#)  [Systems](#)  page and included in this formation.

## Rules

When creating *Data Ingestion for Industry Cloud Solutions* formations, keep in mind the following rules:

- Only systems that are part of the industry cloud solutions, support data ingestion, and are provisioned from SAP for Me can be included in formations of type *Data Ingestion for Industry Cloud Solutions*.
- System that is part of the industry cloud solutions published by SAP are included in at most one *Data Ingestion for Industry Cloud Solutions* formation when creating this formation.
- System of type *Data Ingestion for Industry Cloud Solutions* is automatically registered and included in the *Data Ingestion for Industry Cloud Solutions* formation.
- System of type *Data Ingestion for Industry Cloud Solutions* cannot be excluded from the *Data Ingestion for Industry Cloud Solutions* formation. Once you delete this formation, the *Data Ingestion for Industry Cloud Solutions* system is automatically deleted.

## Prerequisites

You are a global account administrator, or you are a system landscape administrator. See [Working with Role Collections \[page 2274\]](#).

## Procedure

1. In the *System Landscape* *Systems* page of the SAP BTP cockpit, browse the already added systems that are part of the industry cloud solutions published by SAP in your customer system landscape.
2. In the *System Landscape* *Formations* page, create a formation of type *Data Ingestion for Industry Cloud Solutions* and include the relevant systems in it.
  1. Add any name that helps you identify your formation.
  2. In the *Formation Type* dropdown menu, select *Data Ingestion for Industry Cloud Solutions*.
  3. Select the systems that you want to include in the formation.
  4. Review your selections and create the formation.

## Related Information

[Data Integration Overview](#)

## 5.4.3 Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment

Extend SAP S/4HANA Cloud with extension applications running on the cloud platform using automated integration configuration.

### Overview

#### ⓘ Note

The SAP S/4HANA Cloud Extensibility service is available for the EU-only access regions. See [Regions](#).

#### ⓘ Note

This documentation refers to SAP S/4HANA Cloud Public Edition. See [Introduction to the Universe of SAP S/4HANA Cloud Public Edition](#).

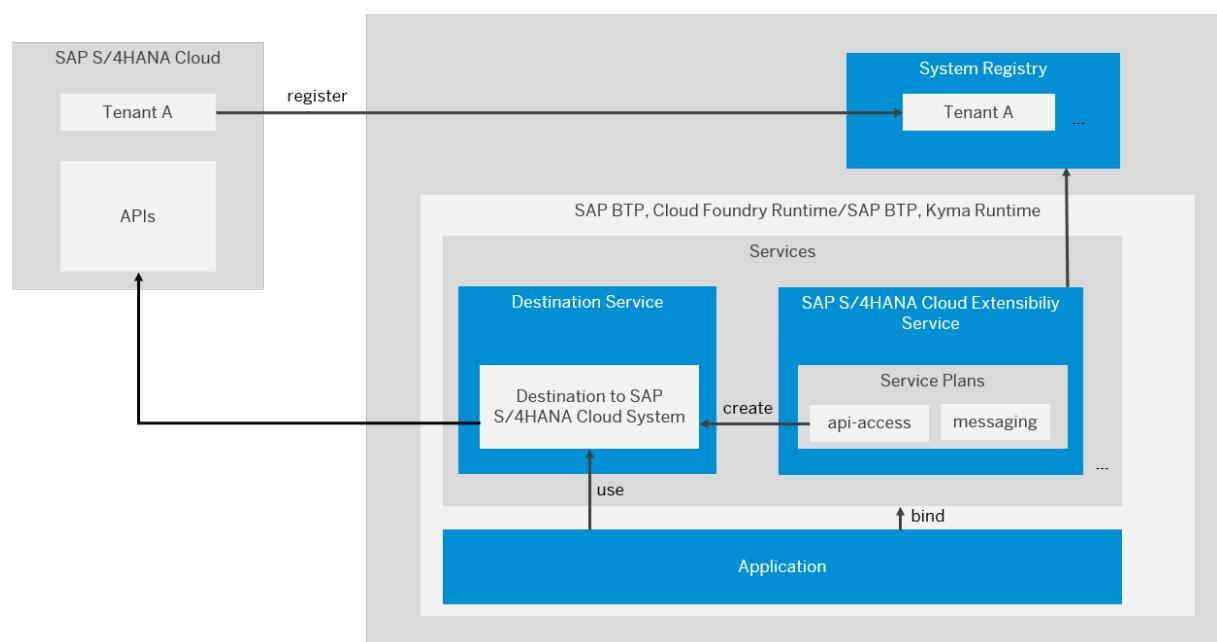
The extension capabilities of SAP BTP offer a standard way for extending SAP S/4HANA Cloud and developing event-driven extensions and applications.

You can extend SAP S/4HANA Cloud without disrupting its performance and core processes. When building extension applications, you can also benefit from the automation of the integration between the cloud platform and SAP S/4HANA Cloud.

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

The following graphic provides a high-level overview of the integration between the cloud platform and SAP S/4HANA Cloud:



## Process Flow

To integrate the cloud platform and SAP S/4HANA Cloud so that you can build extension applications, you need to:

Integrating SAP BTP and SAP S/4HANA Cloud

Process Step	Related Documentation
1. Connect the SAP S/4HANA Cloud system you want to extend with the corresponding global account in SAP BTP.	<a href="#">Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP [page 2002]</a>
During the pairing process you create an registration token which is then used by the SAP S/4HANA Cloud system tenant administrator to configure the integration on the SAP S/4HANA Cloud system side.	<p><b>① Note</b></p> <p>You cannot migrate the registered SAP S/4HANA Cloud systems between global accounts.</p> <p>If you want to start using another global account, you will have to register your SAP S/4HANA Cloud systems again.</p>
2. Make the SAP S/4HANA Cloud system accessible in the SAP BTP subaccounts in which you want to build your extension applications.	<a href="#">Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service [page 2049]</a>
To do so, you configure the entitlements and assign the corresponding quota and service plans to the subaccounts where the extension applications will reside for the system you registered in the previous step.	

Process Step	Related Documentation
<p>3. Configure the communication flow.</p> <p>You have the following options:</p> <ul style="list-style-type: none"><li>Consume the SAP S/4HANA Cloud APIs (inbound connection) or consume APIs exposed by the extension application from SAPS/4 HANA Cloud (outbound connection)</li></ul> <p>To do so, you create a service instance of the <a href="#">SAP S/4HANA Cloud Extensibility</a> service using the <i>api-access</i> service plan.</p> <p>For communication arrangements with inbound connections, an HTTP destination on a subaccount level is automatically generated in this subaccount during the service instance creation. It contains all instance binding properties which are sufficient to establish connection to the SAP S/4HANA Cloud system. When creating the service instance, you configure the communication arrangement and the authentication type for the connection.</p> <p>The following authentication scenarios for SAP S/4HANA Cloud are supported:</p> <ul style="list-style-type: none"><li>Basic Authentication (inbound and outbound connections)</li><li>OAuth 2.0 SAML Bearer Assertion (inbound connections)</li><li>OAuth 2.0 Client Credentials (outbound Connections)</li><li>No Authentication (outbound connections)</li></ul> <p>Both predefined and custom communication scenarios are supported.</p> <ul style="list-style-type: none"><li>Enable the consumption of SAP S/4HANA Cloud events.</li></ul> <p>If you want to create event-based extensions for SAP S/4HANA Cloud using the SAP Event Mesh, you have to create a service instance of the <a href="#">SAP S/4HANA Cloud Extensibility</a> service using the <i>messaging</i> service plan.</p> <ul style="list-style-type: none"><li>A combination of both</li></ul>	<ul style="list-style-type: none"><li><a href="#">Creating a Service Instance to Consume the SAP S/4HANA Cloud APIs [page 2050]</a></li><li><a href="#">Enable the Consumption of SAP S/4HANA Cloud Events [page 2072]</a></li></ul>

## 5.4.3.1 Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service

Configure the required entitlements to make the APIs of the registered SAP S/4HANA Cloud system accessible in your subaccount in which your extension applications will reside.

### Prerequisites

- You are an administrator of the global account in SAP BTP.
- The subaccount is in the SAP BTP, Cloud Foundry environment with enabled Cloud Foundry, or Kyma, or both capabilities.
- You have registered an SAP S/4HANA Cloud system. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

### Context

An entitlement is your right to provision and consume a resource. In other words, the entitlement is the respective service plan, *api-access* or *messaging*, that you're entitled to use.

### Procedure

1. In the SAP BTP cockpit, navigate to your global account.
2. In the navigation area, choose *Entitlements* *Entity Assignments*.
3. On the *Entity Assignments* screen, select your subaccount in the *Subaccounts/Directories* field.
4. Choose *Edit*, and then choose *Add Service Plans*.
5. Select the *SAP S/4HANA Cloud Extensibility* service.

#### Note

To have the *SAP S/4HANA Cloud Extensibility* service in the list, you need to have registered at least one SAP S/4HANA Cloud system.

6. In the *Service Details* area, select the system you have registered. In the *Available Plans* area, select the required service plans, and then choose *Add Service Plan*.

#### Note

For more information about the available service plans for SAP S/4HANA Cloud Extensibility service, see [Supported Service Plans for SAP S/4HANA Cloud \[page 2101\]](#).

7. Choose *Save*.

## Related Information

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

### 5.4.3.2 Create a Service Instance to Consume the SAP S/4HANA Cloud APIs

To enable the integration of your extension applications with the SAP S/4HANA Cloud system you have registered in the SAP BTP global account, and to configure the communication flow, you create a service instance of the SAP S/4HANA Cloud Extensibility service.

## Prerequisites

For a communication flow with SAML Bearer Assertion authentication, you must:

- Configure Single-Sign On (SSO). See [Single Sign-On Configuration](#).
- Protect your application. See [Configure Application Authentication](#).

## Context

In both Cloud Foundry and Kyma environments, you consume services by creating a service instance. Service instances are created using a specific service plan.

To allow applications running on SAP BTP to consume SAP S/4HANA Cloud APIs, you need to create a service instance of the SAP S/4HANA Cloud Extensibility service using the [api-access](#) service plan.

### ⓘ Note

These service plans have been deprecated:

- [sap\\_com\\_0109](#)
- [sap\\_com\\_0009](#)
- [sap\\_com\\_0008](#)

However, you can still enable these communication scenarios using the [api-access](#) service plan:

- *Sales Order Integration (SAP\_COM\_0109)*: allows you to integrate your extension applications with sales order processing in SAP S/4HANA Cloud. For more information, see [https://api.sap.com/api/API\\_SALES\\_ORDER\\_SRV/overview](https://api.sap.com/api/API_SALES_ORDER_SRV/overview).
- *Product Integration (SAP\_COM\_0009)*: enables you to replicate product master data from client system to SAP S/4HANA system. For more information, see <https://api.sap.com/api/PRODUCTMDMBULKREPLICAREQUEST/overview>.
- *Business Partner, Customer and Supplier Integration (SAP\_COM\_0008)*: allows you to consume the Business Partner API which enables you to create, read, update, and delete master data related to

Business Partners, Suppliers, and Customers in an SAP S/4HANA system. For more information, see [https://api.sap.com/api/API\\_BUSINESS\\_PARTNER/overview](https://api.sap.com/api/API_BUSINESS_PARTNER/overview).

For a sample JSON for these communication scenarios, see [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

The [api-access](#) service plan define the access to the corresponding SAP S/4HANA Cloud APIs. It supports both predefined and custom communication scenarios for consuming the SAP S/4HANA Cloud APIs and integrating your extension applications. See:

- [Supported Service Plans for SAP S/4HANA Cloud \[page 2101\]](#)
- [Custom Communication Scenarios](#)

You create the service instance in your subaccount with the respective environment enabled. When creating the service instance, you configure the connectivity by specifying the required configurations in a JSON format. The following authentication scenarios are supported for the communication flow between the extension application and SAP S/4HANA Cloud:

- Basic Authentication (inbound and outbound connections)
- OAuth 2.0 Client Credentials (outbound connections)
- OAuth 2.0 SAML Bearer Assertion (inbound connections)

To communicate with SAP S/4HANA Cloud the extension application can use Principal Propagation which is done using OAuth 2.0 SAML Bearer Assertion flows. Principal Propagation means you forward the identity of the logged-in cloud users when accessing or updating data in the SAP S/4HANA Cloud system. This is useful in scenarios where you need to have restricted data access based on the logged-in user from your extension. Or, you want to ensure only users with the right permissions are able to update the system via extensions deployed in SAP BTP.

To use this authentication scenario, you first need to configure single-sign on (SSO) with the Identity Authentication service and protect your application. See [Single Sign-On Configuration](#).

- Client Certificate Authentication (inbound and outbound connections)
- OAuth2mTLS (outbound connections)

Depending on whether you are using Cloud Foundry or Kyma environment, you have to follow different steps to create an SAP S/4HANA Cloud Extensibility service instance:

- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Cloud Foundry Environment \[page 2052\]](#)
- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2054\]](#)

## Related Information

[Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#)

## 5.4.3.2.1 Create an SAP S/4HANA Cloud Extensibility Service Instance in the Cloud Foundry Environment

### Prerequisites

- Before creating an SAP S/4HANA Cloud Extensibility service instance in the Cloud Foundry environment, see [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 2050\]](#).
- Have enabled Cloud Foundry environment for your subaccount. See [Enable Environment or Create Environment Instance \[page 2179\]](#).
- Have registered an SAP S/4HANA Cloud system. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).
- Have configured the entitlements to the SAP S/4HANA Cloud Extensibility service. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

### Context

To consume the SAP S/4HANA Cloud APIs (inbound connection) or consume APIs exposed by the extension application from SAP S/4HANA Cloud (outbound connection), you create an SAP S/4HANA Cloud Extensibility service instance. When creating the service instance, you configure the communication arrangement and the authentication type for the connection.

For communication arrangements with inbound connections, a destination on a subaccount level with the same name as the service instance name is automatically created in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP S/4HANA Cloud system.

#### ⓘ Note

Make sure that you don't already have a destination with the same name as the service instance. If you do, you will not be able to create the service instance.

#### ⓘ Note

Multitenant applications, that this subaccount is subscribed to, can access the destination and connect to the SAP S/4HANA Cloud system without binding to the created service instance.

See [Developing Multitenant Business Applications in the Cloud Foundry Environment](#).

## Procedure

1. In the SAP BTP cockpit, navigate to the subaccount in which you want to create a service instance.
2. In the navigation area, choose   *Services*  *Service Marketplace* .
3. All services available to you appear.
4. To enable the integration with an SAP S/4HANA Cloud system that you have registered in SAP BTP global account, choose *SAP S/4HANA Cloud Extensibility*.
5. In the *SAP S/4HANA Cloud Extensibility* page, choose *Create*.
6. In the *New Instance or Subscription* wizard:
  - a. In the *Service* dropdown list, ensure you have selected the *SAP S/4HANA Cloud Extensibility* service.
  - b. In the *Plan* dropdown list, select the *api-access* service plan.
  - c. In the *Runtime Environment* dropdown list, select *Cloud Foundry*.
  - d. In the *Space* dropdown list, select your space. If you haven't create a space yet, you can do it at this point.
  - e. In the *System Name* dropdown list, select the SAP S/4HANA Cloud system you have registered.
  - f. In the *Instance Name* field, enter a name for your instance. Choose *Next*.
  - g. To define the communication arrangement and the authentication type for the API access, specify a JSON file or specify parameters in the JSON format. If you decide to define the communication arrangement and the authentication type later on, you have to delete this service instance and create it again. Choose *Next*.

For more information about the structure of the JSON file, see [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

- h. Choose *Create*.

After you have created the service instance:

- The newly created instance appears in the list of instances in the *Instance and Subscriptions* page.
- An HTTP destination on a subaccount level with the same name as the service instance name is automatically generated in this subaccount.

Alternatively, you can use the Cloud Foundry Command Line Interface (cf CLI) to create the service instance using the technical name of the SAP S/4HANA Cloud Extensibility service which is **s4-hana-cloud**.

For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

### Note

You can use the cf CLI to troubleshoot if the creation of the service instance fails. To do that, use this command in the cf CLI:

```
cf service <service_instance_name>
```

## Next Steps

After you have created the service instance:

- The newly created instance appears in the list of instances in the [Instance and Subscriptions](#) page.
- An HTTP destination on a subaccount level with the same name as the service instance name is automatically generated in this subaccount.

Alternatively, you can use the Cloud Foundry Command Line Interface (cf CLI) to create the service instance using the technical name of the SAP S/4HANA Cloud Extensibility service which is **s4-hana-cloud**.

For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

#### Note

You can use the cf CLI to troubleshoot if the creation of the service instance fails. To do that, use this command in the cf CLI:

```
cf service <service_instance_name>
```

After creating the [SAP S/4HANA Cloud Extensibility](#) service instance, you have the following options for configuring the connectivity for your extension application:

- Bind the instance to an application, and it will be assigned an access URL and credentials to the corresponding API. For more information about binding applications to service instances, see [Binding Service Instances to Applications](#) in the SAP BTP documentation.
- Consume the automatically generated destination.

To consume the destination, you use the destination service. You can either consume the Destination service directly, or configure the application router to consume it.

#### Note

The name of the destination is the same as the name of the service instance you have created.

- For more information about consuming the destination service using the application router, see [Application Routes and Destinations](#).
- For more information about consuming the destination service directly, see [Consuming the Destination Service \(Cloud Foundry Environment\)](#).

### 5.4.3.2.2 Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment

#### Prerequisites

- Before creating an SAP S/4HANA Cloud Extensibility service instance in the Kyma environment, see [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 2050\]](#).
- Have the Kyma environment enabled for the subaccount you are using. See [Create a Kyma Instance \[page 2992\]](#).
- Configure the roles in the Kyma environment. See [Assign Roles in the Kyma Environment \[page 3137\]](#).

- The SAP BTP Operator module is enabled. See [Add and Delete a Kyma Module \[page 3012\]](#)
- Have the entitlements of the SAP S/4HANA Cloud Extensibility service configured. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

## Context

To consume the SAP S/4HANA Cloud APIs (inbound connection) or consume APIs exposed by the extension application from SAP S/4HANA Cloud (outbound connection), you create an SAP S/4HANA Cloud Extensibility service instance. When creating the service instance, you configure the communication arrangement and the authentication type for the connection.

For communication arrangements with inbound connections, a destination on a subaccount level with the same name as the service instance name is automatically created in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP S/4HANA Cloud system.

### Note

Make sure that you don't already have a destination with the same name as the service instance. If you do, you will not be able to create the service instance.

## Procedure

1. Navigate to the subaccount for which you want to create an SAP S/4HANA Cloud Extensibility service instance.
2. On the subaccount [Overview](#) page, in the *Kyma Environment* section, open the Kyma dashboard.
3. Choose [Namespaces](#) from the left-hand side navigation and open the namespace in which you want to create a service instance.
4. Choose  [Service Instances](#)  from the left-hand side navigation.
5. In the [Service Instances](#) page, choose [Create Service Instance](#) in the upper right-hand corner. A new dialog opens.
6. Choose the [Simple](#) tab and fill in the following fields:
  - Give a meaningful name of the new SAP S/4HANA Cloud Extensibility service instance.
  - In the *Offering Name* field, enter **s4-hana-cloud**, which is the technical name of the SAP S/4HANA Cloud Extensibility service.
  - In the *Plan Name* field, enter **api-access**.
7. Choose the [YAML](#) tab.

To define the communication arrangement and the authentication type for the API access, specify in *parameters*: in the *spec:* section, the parameters listed in this YAML file: [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

For specific examples, see [Communication Arrangement YAML File - Examples \[page 2068\]](#).

8. Choose [Create](#).

## Next Steps

After creating the [SAP S/4HANA Cloud Extensibility](#) service instance, you must bind the instance to an application, and it will be assigned an access URL and credentials to the corresponding API. See [Using SAP BTP Services in the Kyma Environment](#) [page 1975].

### 5.4.3.2.3 Communication Arrangement JSON/YAML File - Properties

Learn how to construct a JSON or YAML file for the SAP S/4HANA Cloud APIs, including parameters such as system name, communication arrangement, authentication type, communication system, outbound services, and additional properties. It also provides guidelines and rules for each property.

#### Context

To construct the JSON file for the Cloud Foundry environment or the YAML file for the Kyma environment for the SAP S/4HANA Cloud APIs you want to use, you can use these parameters. To access the specific documentation of these APIs, see [SAP S/4HANA Cloud APIs at SAP API Business Hub](#).

The information that you need to construct the JSON or the YAML file is available in the Display Communication Scenario application in the corresponding SAP S/4HANA Cloud system. It contains information such as scenario details and properties, and supported inbound and outbound authentication methods. See [Display Communication Scenarios](#).

For specific sample JSON files, see [Communication Arrangement JSON File - Examples](#) [page 2064].

For specific sample YAML files, see [Communication Arrangement YAML File - Examples](#) [page 2068].

#### Properties

Parameter	Description
systemName	<p>The name of the system you have registered in SAP BTP global account.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p><b>① Note</b></p><p>The system must be in status <i>Registered</i>.</p></div> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>Required: Yes</li></ul>

Parameter	Description
communicationArrangement	<p>Represents a communication arrangement in SAP S/4HANA Cloud.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes</li> </ul>
communicationArrangementName	<p>A communicationArrangement property.</p> <p>Meaningful name of the communication arrangement that will be created for the SAP S/4HANA Cloud tenant.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Allowed characters: [a-zA-Z0-9_-]</li> <li>Max length: 80</li> <li>Required: Yes</li> </ul>
scenarioId	<p>A communicationArrangement property.</p> <p>The ID of the SAP S/4HANA Cloud communication scenario.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes</li> <li>Allowed characters: [A-Z0-9_]</li> <li>Pattern: <b>SAP_COM_&lt;number&gt;</b></li> </ul>

Parameter	Description
inboundAuthentication	<p>A communicationArrangement property.</p> <p>The authentication type for the SAP S/4HANA Cloud API access.</p> <p><b>ⓘ Note</b></p> <p>Currently, the following authentication methods are supported:</p> <ul style="list-style-type: none"><li>• Basic Authentication</li><li>• SAML Bearer Assertion Authentication</li></ul> <p><b>ⓘ Note</b></p> <p>The generated X.509 certificate that is used to configure the trust between the cloud platform and SAP S/4HANA Cloud system expires two years after you have created the instance.</p> <ul style="list-style-type: none"><li>• Client Certificate Authentication</li></ul>

Parameter	Description
outboundAuthentication	<p>A communicationArrangement property.</p> <p>The type of the authentication used by SAP S/4HANA Cloud to call SAP BTP APIs.</p> <div data-bbox="822 496 927 530" style="background-color: #e0f2ff; padding: 5px;"><b> ⓘ Note</b></div> <p>Currently, the following authentication methods are supported:</p> <ul style="list-style-type: none"> <li>• Basic Authentication</li> <li>• OAuth 2.0 Client Credentials</li> <li>• No Authentication</li> <li>• OAuth2mTLS</li> <li>• Client Certificate Authentication</li> </ul>
communicationSystem	<p>A communicationArrangement property.</p> <p>This represents the <i>Communication System</i> view of the communication arrangement.</p>
communicationSystemHostname	<p>A communicationSystem property.</p> <p>The URL of the remote system hosting the APIs that will be consumed in case the scenario contains outbound communication. For SAP S/4HANA Cloud extensions on SAP BTP, this is the URL of the extension application running on SAP BTP.</p> <p>This is equivalent to  <a href="#">Technical Data</a>  <a href="#">General</a>  <a href="#">Host Name</a>  in the <i>Communication System</i> view.</p>
Rules/Guidelines	<ul style="list-style-type: none"> <li>• Required: Yes, if communicationArrangement is specified.</li> </ul>

Parameter	Description
port	<p>A communicationSystem property.</p> <p>The port for outbound calls to the remote system hosting the APIs that will be consumed in case the scenario contains outbound communication.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No If not specified, the default <a href="#">443</a> port is used for the communication.</li> <li>• Type: String</li> <li>• Allowed values: positive integer [<a href="#">1-65535</a>]</li> </ul>
oAuthAuthEndpoint	<p>A communicationSystem property.</p> <p>The OAuth authorization endpoint of the remote OAuth service in case the communication scenario contains outbound communication. This is equivalent to  <a href="#">Technical Data</a>  <a href="#">OAuth 2.0 Settings</a>  in the <a href="#">Communication System</a> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
oAuthTokenEndpoint	<p>A communicationSystem property.</p> <p>The OAuth token endpoint. This is equivalent to  <a href="#">Technical Data</a>  <a href="#">OAuth 2.0 Settings</a>  in the <a href="#">Communication System</a> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
outboundCommunicationUser	<p>A communicationSystem property.</p> <p>The communication user used for outbound authentication. This is equivalent to an entry under the <a href="#">Users for Outbound Communication</a> table in the <a href="#">Communication System</a> view in the SAP S/4HANA Cloud system.</p> <div style="background-color: #f0f8ff; padding: 10px;"> <p> <b>Note</b></p> <p>Currently, only users with authentication method <a href="#">User Name and Password</a> are supported.</p> </div> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>

Parameter	Description
username	<p>An <code>outboundCommunicationUser</code> property. The username of the communication user.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if the <code>outboundCommunicationUser</code> property is specified.</li> </ul>
password	<p>An <code>outboundCommunicationUser</code> property. The password of the communication user.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if the <code>outboundCommunicationUser</code> property is specified.</li> </ul>
outboundServices	<p>A <code>communicationArrangement</code> property. A list of outbound service objects. This is equivalent to the <i>Outbound Services</i> section in the SAP S/4HANA Cloud <i>Communication Arrangement</i> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>
outboundService	<p>An <code>outboundServices</code> object. A specific outbound service object.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>
id	<p>An <code>outboundService</code> property. The id of the outbound service. It must match the <i>Outbound Service ID</i> displayed in the SAP S/4HANA Cloud system.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if the <code>outboundServices</code> property is specified.</li> </ul> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>When you specify the <code>id</code> property of the outbound service, you can skip the <code>name</code> property.</p> </div>

Parameter	Description
name	<p>An outboundService property.</p> <p>The name of the outbound service. It must be an exact match of the name displayed in the SAP S/4HANA Cloud system.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if the <code>outboundServices</code> property is specified, and the <code>id</code> property is not specified.</li> </ul> <div style="background-color: #f0f8ff; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>If the <code>id</code> property is specified, the <code>name</code> property is not required.</p> </div>
urlPath	<p>An outboundService property.</p> <p>This is equivalent to the <i>Path</i> field in the SAP S/4HANA Cloud system. It is used to configure the API path in the outbound service URL.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>
isServiceActive	<p>An outboundService property.</p> <p>This is equivalent to the <i>Service Status</i> checkbox in the SAP S/4HANA Cloud system. In some communication scenarios, some outbound services must be disabled. You can achieve that by setting this parameter to <code>false</code>.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Default value: <code>true</code></li> </ul>
attributes	<p>An outboundService parameter.</p> <p>A list of attribute objects.</p> <p>This is equivalent to the <i>Additional Properties</i> section of the outbound service.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>

Parameter	Description
attribute	<p>An <code>attributes</code> object.</p> <p>A specific attribute object. Represents an additional property in the outbound service.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
name	<p>An <code>attribute</code> property.</p> <p>The name of the additional property of the outbound service. It is equivalent to the <a href="#">Technical Property Name</a> column in the <a href="#">Additional Properties</a> section in the Display Communication Scenarios app in the SAP S/4HANA Cloud system. See <a href="#">Display Communication Scenarios</a>.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes, if an <code>attribute</code> object is specified.</li> </ul>
value	<p>An <code>attribute</code> property.</p> <p>Enter value for the additional property of the outbound service.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes, if an <code>attribute</code> object is specified.</li> </ul>
attributes	<p>A <code>communicationArrangement</code> parameter.</p> <p>A list of attribute objects.</p> <p>This is equivalent to the <a href="#">Additional Properties</a> section of the communication arrangement in SAP S/4HANA Cloud.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
attribute	<p>An <code>attributes</code> object.</p> <p>A specific attribute object. Represents an additional property in the communication arrangement.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>

Parameter	Description
name	<p>An attribute property.</p> <p>It is equivalent to the <i>Technical Property Name</i> section in the <i>Additional Properties</i> table in the Display Communication Scenarios application in the SAP S/4HANA Cloud system. See <a href="#">Display Communication Scenarios</a>.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if an attribute object is specified for communicationArrangement.</li> </ul>
value	<p>An attribute property.</p> <p>Enter a value for the additional property of the communication arrangement.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes, if an attribute object is specified for communicationArrangement.</li> </ul>

## Related Information

[Communication Arrangement JSON File - Examples \[page 2064\]](#)

[Communication Arrangement YAML File - Examples \[page 2068\]](#)

### 5.4.3.2.4 Communication Arrangement JSON File - Examples

The examples in this section will help you to create the service JSON descriptor used for defining the communication arrangement and the authentication type for the SAP S/4HANA Cloud API access.

The information that you need to create the JSON file is available in the *Display Communication Scenario* app in the corresponding SAP S/4HANA Cloud system. It contains information such as scenario details and properties, and supported inbound and outbound authentication methods. See [Display Communication Scenarios](#).

#### ⓘ Note

The purpose of these examples is just to give you an idea how you construct your JSON file. For more details of the properties of these JSON files, see [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

## Example of Enabling Communication Scenario of Type Basic Authentication

This is an example of a JSON file for a communication arrangement with an inbound connection with *Basic Authentication* and an outbound connection with *Basic Authentication*.

```
{
  "systemName": "DEMO",
  "communicationArrangement": {
    "communicationArrangementName": "INBOUND_COMMUNICATION_ARRANGEMENT",
    "scenarioId": "SAP_COM_0008",
    "inboundAuthentication": "BasicAuthentication",
    "outboundAuthentication": "BasicAuthentication",
    "outboundServices": [
      {
        "id": "DEBMAS_IDOC",
        "isServiceActive": false
      },
      {
        "id": "CREMAS_IDOC",
        "isServiceActive": false
      },
      {
        "id": "ADRMAS_IDOC",
        "isServiceActive": false
      },
      {
        "id": "ADR3MAS_IDOC",
        "isServiceActive": false
      },
      {
        "id": "ADR2MAS_IDOC",
        "isServiceActive": false
      },
      {
        "id": "CO_MDG_BP_RPLCTRQ_SPRX",
        "isServiceActive": false
      },
      {
        "id": "CO_MDG_BP_RELATIONSHIP_OUT_SPRX",
        "isServiceActive": false
      },
      {
        "id": "CO_MDG_BP_RPLCTCO_SPRX",
        "isServiceActive": false
      },
      {
        "id": "CO_MDG_BP_RELATIONSHIP_CNF_OUT_SPRX",
        "isServiceActive": false
      }
    ],
    "communicationSystem": {
      "communicationSystemHostname": "default.com",
      "outboundCommunicationUser": {
        "username": "DefaultUser",
        "password": "DefaultPassword"
      }
    }
  }
}
```

## Example of Enabling Communication Scenario of Type OAuth2ClientCredentials

This is an example of a JSON file for a communication arrangement with an outbound connection with authentication type *OAuth2ClientCredentials*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "outboundAuthentication": "OAuth2ClientCredentials",  
        "communicationArrangementName": "0219_ARRANGEMENT",  
        "scenarioId": "SAP_COM_0219",  
        "communicationSystem": {  
            "communicationSystemHostname": "default.com",  
            "oAuthAuthEndpoint": "oauth.com/oauth/authorize",  
            "oAuthTokenEndpoint": "oauth.com/oauth/token",  
            "outboundCommunicationUser": {  
                "username": "DefaultUser",  
                "password": "DefaultPassword"  
            }  
        }  
    }  
}
```

## Example for Enabling Communication Scenario of Type OAuth2SAMLBearerAssertion

This is an example of a JSON file for a communication arrangement with an inbound connection with *OAuth2SAMLBearerAssertion* and an outbound connection with *NoAuthentication*.

To communicate with SAP S/4HANA Cloud the extension application can use Principal Propagation which is done using OAuth 2.0 SAML Bearer Assertion flows. Principal Propagation means you forward the identity of the logged-in cloud users when accessing or updating data in the SAP S/4HANA Cloud system.

This is useful in scenarios where you need to have restricted data access based on the logged-in user from your extension. Or, you want to ensure only users with the right permissions are able to update the system via extensions deployed in SAP BTP, Cloud Foundry runtime.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "scenarioId": "SAP_COM_0213",  
        "communicationArrangementName": "0213_ARRANGEMENT",  
        "inboundAuthentication": "OAuth2SAMLBearerAssertion",  
        "communicationSystem": {  
            "communicationSystemHostname": "default.com"  
        }  
    }  
}
```

## Example for Enabling Communication Scenario of Type NoAuthentication

This is an example of a JSON file for a communication arrangement with an outbound connection with authentication type *NoAuthentication*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "outboundAuthentication": "NoAuthentication",  
        "communicationArrangementName": "0215_ARRANGEMENT",  
        "scenarioId": "SAP_COM_0215",  
        "outboundServices": [  
            {  
                "id": "SAP_COM_0215_0001_REST",  
                "isServiceActive": true  
            }  
        ],  
        "communicationSystem": {  
            "communicationSystemHostname": "default.com"  
        }  
    }  
}
```

## Example for Enabling Communication Scenario of Type Client Certificate Authentication with Inbound Connection

This is an example of a JSON file for a communication arrangement with an inbound connection with authentication type *ClientCertificateAuthentication*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "scenarioId": "SAP_COM_0724",  
        "communicationArrangementName": "CommunicationArrangementName",  
        "inboundAuthentication": "ClientCertificateAuthentication"  
    }  
}
```

## Example for Enabling Communication Scenario of Type Client Certificate Authentication with Outbound Connection

This is an example of a JSON file for a communication arrangement with an outbound connection with authentication type *ClientCertificateAuthentication*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "communicationArrangementName": "CommunicationArrangementName",  
        "scenarioId": "SAP_COM_0047",  
        "outboundAuthentication": "ClientCertificateAuthentication",  
        "communicationSystem": {  
            "communicationSystemHostname": "default.com"  
        }  
    }  
}
```

```
        }
    }
}
```

## Example for Enabling Communication Scenario of Type OAuth2mTLS

This is an example of a JSON file for a communication arrangement with an outbound connection with authentication type *OAuth2mTLS*.

```
{
  "systemName": "DEMO",
  "communicationArrangement": {
    "outboundAuthentication": "OAuth2mTLS",
    "communicationArrangementName": "CommunicationArrangementName",
    "scenarioId": "SAP_COM_0080",
    "communicationSystem": {
      "communicationSystemHostname": "default.com",
      "oAuthAuthEndpoint": "oauth.com/oauth/authorize",
      "oAuthTokenEndpoint": "oauth.com/oauth/token",
      "outboundCommunicationUser": {
        "username": "DefaultUser"
      }
    }
  }
}
```

### 5.4.3.2.5 Communication Arrangement YAML File - Examples

The examples in this section will help you to create the service YAML descriptor used for defining the communication arrangement and the authentication type for the SAP S/4HANA Cloud API access.

The information that you need to create the YAML file is available in the [Display Communication Scenario](#) app in the corresponding SAP S/4HANA Cloud system. It contains information such as scenario details and properties, and supported inbound and outbound authentication methods. See [Display Communication Scenarios](#).

#### ⓘ Note

The purpose of these examples is just to give you an idea how you construct your YAML file. For more details of the properties of these YAML files, see [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

## Example of Enabling Communication Scenario of Type Basic Authentication

This is an example of a YAML file for a communication arrangement with an inbound connection with *Basic Authentication* and an outbound connection with *Basic Authentication*.

```
spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
parameters:
  systemName: DEMO
  communicationArrangement:
    communicationArrangementName: INBOUND_COMMUNICATION_ARRANGEMENT
    scenarioId: SAP_COM_0008
    inboundAuthentication: BasicAuthentication
    outboundAuthentication: BasicAuthentication
    outboundServices:
      - id: DEBMAS_IDOC
        isServiceActive: false
      - id: CREMAS_IDOC
        isServiceActive: false
      - id: ADRMAS_IDOC
        isServiceActive: false
      - id: ADR3MAS_IDOC
        isServiceActive: false
      - id: ADR2MAS_IDOC
        isServiceActive: false
      - id: CO_MDG_BP_RPLCTRQ_SPRX
        isServiceActive: false
      - id: CO_MDG_BP_RELATIONSHIP_OUT_SPRX
        isServiceActive: false
      - id: CO_MDG_BP_RPLCTCO_SPRX
        isServiceActive: false
      - id: CO_MDG_BP_RELATIONSHIP_CNF_OUT_SPRX
        isServiceActive: false
  communicationSystem:
    communicationSystemHostname: default.com
    outboundCommunicationUser:
      username: DefaultUser
      password: DefaultPassword
```

## Example of Enabling Communication Scenario of Type OAuth2ClientCredentials

This is an example of a YAML file for a communication arrangement with an outbound connection with authentication type *OAuth2ClientCredentials*.

```
spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
parameters:
  systemName: DEMO
  communicationArrangement:
    outboundAuthentication: OAuth2ClientCredentials
    communicationArrangementName: 0219_ARRANGEMENT
    scenarioId: SAP_COM_0219
  communicationSystem:
    communicationSystemHostname: default.com
```

```

oAuthAuthEndpoint: oauth.com/oauth/authorize
oAuthTokenEndpoint: oauth.com/oauth/token
outboundCommunicationUser:
  username: DefaultUser
  password: DefaultPassword

```

## Example for Enabling Communication Scenario of Type OAuth2SAMLBearerAssertion

This is an example of a YAML file for a communication arrangement with an inbound connection with `OAuth2SAMLBearerAssertion` and an outbound connection with `NoAuthentication`.

To communicate with SAP S/4HANA Cloud the extension application can use Principal Propagation which is done using OAuth 2.0 SAML Bearer Assertion flows. Principal Propagation means you forward the identity of the logged-in cloud users when accessing or updating data in the SAP S/4HANA Cloud system.

This is useful in scenarios where you need to have restricted data access based on the logged-in user from your extension. Or, you want to ensure only users with the right permissions are able to update the system via extensions deployed in SAP BTP, Kyma runtime.

```

spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
parameters:
  systemName: DEMO
  communicationArrangement:
    communicationArrangementName: 0213_ARRANGEMENT
    scenarioId: SAP_COM_0213
    inboundAuthentication: OAuth2SAMLBearerAssertion
    communicationSystem:
      communicationSystemHostname: default.com

```

## Example for Enabling Communication Scenario of Type NoAuthentication

This is an example of a YAML file for a communication arrangement with an outbound connection with authentication type `NoAuthentication`.

```

spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
parameters:
  systemName: DEMO
  communicationArrangement:
    communicationArrangementName: 0215_ARRANGEMENT
    scenarioId: SAP_COM_0215
    outboundAuthentication: NoAuthentication
    outboundServices:
      - id: SAP_COM_0215_0001_REST
        isServiceActive: true
    communicationSystem:
      communicationSystemHostname: default.com

```

## Example for Enabling Communication Scenario of Type Client Certificate Authentication with Inbound Connection

This is an example of a YAML file for a communication arrangement with an inbound connection with authentication type *ClientCertificateAuthentication*.

```
spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
  parameters:
    systemName: DEMO
    communicationArrangement:
      communicationArrangementName: CommunicationArrangementName
      scenarioId: SAP_COM_0724
      inboundAuthentication: ClientCertificateAuthentication
```

## Example for Enabling Communication Scenario of Type Client Certificate Authentication with Outbound Connection

This is an example of a YAML file for a communication arrangement with an outbound connection with authentication type *ClientCertificateAuthentication*.

```
spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
  parameters:
    systemName: DEMO
    communicationArrangement:
      communicationArrangementName: CommunicationArrangementName
      scenarioId: SAP_COM_0018
      outboundAuthentication: ClientCertificateAuthentication
      outboundServices:
        - id: CO_MSM_PROJECT_ERPCREATE_REQUE_SPRX
          isServiceActive: true
        - id: CO_MSM_PROJECT_ERPUPDATE_REQUE_SPRX
          isServiceActive: false
        - id: CO_MSM_PROJECT_ERPBY_IDQUERY_R_SPRX
          isServiceActive: false
    communicationSystem:
      communicationSystemHostname: default.com
```

## Example for Enabling Communication Scenario of Type OAuth2mTLS

This is an example of a YAML file for a communication arrangement with an outbound connection with authentication type *OAuth2mTLS*.

```
spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: api-access
```

```

parameters:
  systemName: DEMO
  communicationArrangement:
    communicationArrangementName: CommunicationArrangementName
    scenarioId: SAP_COM_0080
    outboundAuthentication: OAuth2mTLS
    outboundServices:
      - id: SAP_COM_0080_0001_REST
        isServiceActive: true
  communicationSystem:
    communicationSystemHostname: default.com
    oAuthAuthEndpoint: oauth.com/oauth/authorize
    oAuthTokenEndpoint: oauth.com/oauth/token
    outboundCommunicationUser:
      username: DefaultUser

```

### 5.4.3.3 Enable the Consumption of SAP S/4HANA Cloud Events

To create event-based extensions for SAP S/4HANA Cloud you need to set up the messaging between the SAP S/4HANA Cloud system and the SAP Event Mesh.

#### Prerequisites

- You have registered an SAP S/4HANA Cloud tenant in the global account in SAP BTP. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).

#### Context

To consume SAP S/4HANA Cloud events, you need to configure the connectivity between SAP Event Mesh and the SAP S/4HANA Cloud tenant. To do so, you need to create and configure service instances for both SAP S/4HANA Cloud Extensibility service and SAP Event Mesh.

When you create a service instance in a subaccount, an event bus for this subaccount is created. All service instances in a subaccount point to the same event bus and are nothing else but clients providing access credentials for the event bus. You configure the clients using JSON descriptors. Each client has its own name which can be configured with the emname parameter in the JSON descriptor file. Thus, the client is identified with a human readable name instead of a GUID. The name of each client must be unique. As all of the clients point to the same event bus, they must have different namespaces in which to publish events. Each namespace must be unique per subaccount. Each client should publish only in its own unique namespace.

#### Communication flow

The SAP S/4HANA Cloud Extensibility service instance is the client which creates the SAP S/4HANA Cloud integration and provides the credentials for the SAP S/4HANA Cloud system so that the SAP S/4HANA Cloud system can send events to that event bus. The purpose of this client is to send messages to a topic. It sends messages to the namespace which was configured while creating the service instance of S/4HANA Cloud

Extensibility service. See [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#)

The SAP Event Mesh service instance is the client which receives the events. This service instance must have a different name and a different namespace than the ones configured for the SAP S/4HANA Cloud extensibility service instance. Since the purpose of this client is to receive messages from a different client the rules have to take care of that. So, for the SAP Event Mesh service instance, you need to configure the subscribe filter to the SAP S/4HANA Cloud Extensibility service instance. See [Define SAP Event Mesh Service Descriptor JSON/YAML File \[page 2093\]](#).

After both instances are created and configured, you can create the topic-to-queue subscription. To do so, you first create a queue in the client that refers to the SAP Event Mesh service instance and then subscribe the topic of the namespace of the SAP S/4HANA Cloud Extensibility service to that queue. See [Manage Queues](#) in the SAP Event Mesh documentation

The service instance of SAP Event Mesh is the client which consumes the message and should be bound to the extension application.

## Procedure

To configure the connectivity between SAP Event Mesh and the SAP S/4HANA Cloud tenant so that SAP S/4HANA Cloud events can be produced and then consumed by applications running on SAP BTP, you need to perform the following tasks:

1. Add the required quotas to your subaccount as follows:
  1. Assign the [messaging](#) SAP S/4HANA Cloud Extensibility service plan to the subaccount in SAP BTP you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

**ⓘ Note**

You can use the service plan on regions where the SAP Event Mesh service is available.

**ⓘ Note**

The [messaging](#) service plan enables the *Enterprise Eventing Integration (SAP\_COM\_0092)* communication scenario which allows you to consume SAP S/4HANA Cloud events and create event-based extensions.

This service plan has been renamed from [sap\\_com\\_0092](#).

2. Assign the required number of SAP Event Mesh instances to the subaccount in SAP BTP that you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for SAP Event Mesh \[page 2074\]](#).
2. Configure the connectivity between SAP Event Mesh and the SAP S/4HANA Cloud tenant, and then configure event topics for the channel inside SAP S/4HANA Cloud tenant. See [Set Up the Connectivity Between Event Mesh and the SAP S/4HANA Cloud Tenant \[page 2075\]](#).
3. Enable the SAP Event Mesh for your subaccount in SAP BTP, create a queue that is specific to your application, and then subscribe this queue to the channel topic that the SAP S/4HANA Cloud tenant uses to produce events. See [Enable the SAP Event Mesh Service for Your Subaccount in SAP BTP \[page 2089\]](#).

## Related Information

[What Is SAP Event Mesh?](#)

### 5.4.3.3.1 Configure the Entitlements for SAP Event Mesh

To be able to consume SAP S/4HANA Cloud events, you need to configure the entitlements for SAP Event Mesh for the subaccount in SAP BTP.

## Context

The [messaging](#) service plan connects SAP S/4HANA Cloud tenant to SAP Event Mesh for the subaccount where the SAP Event Mesh service entitlement is configured. This SAP Event Mesh service instance allows you to consume events from SAP S/4HANA Cloud.

You need to configure the entitlements for the SAP Event Mesh service to create an SAP Event Mesh service instance. Then, you bind this service instance to an application to consume events from this application.

## Procedure

1. In SAP BTP cockpit, navigate to the global account that contains the subaccount in which you want to make your SAP system accessible.
2. In the navigation area, choose  [Entitlements](#)  [Entity Assignments](#).
3. On the [Entity Assignments](#) screen, select your subaccount in the [Subaccounts/Directories](#) field.
4. Choose [Edit](#), and then choose [Add Service Plans](#).
5. Select the [SAP Event Mesh](#) service.
6. In the [Service Details](#) area, select the system you have registered. In the [Available Plans](#) area, select the [default](#) service plan, and then choose [Add Service Plan](#).
7. Choose [Save](#).

## Related Information

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

## 5.4.3.3.2 Set Up the Connectivity Between Event Mesh and the SAP S/4HANA Cloud Tenant

### Overview

To configure the connectivity between the SAP S/4HANA Cloud tenant and SAP Event Mesh and to enable the exchange of credentials between the two systems, you first need to create an [SAP S/4HANA Cloud Extensibility](#) service instance with service plan messaging. For more information about the messaging service plan, see [Supported Service Plans for SAP S/4HANA Cloud \[page 2101\]](#).

When creating this service instance, you create the required configurations in both the SAP S/4HANA Cloud tenant and the Event Mesh system associated with the subaccount in SAP BTP, so that events can flow. Depending on the SAP BTP environment, to create an SAP S/4HANA Cloud Extensibility service instance, you have to choose one of the following options:

- [Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment \[page 2076\]](#)
- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2078\]](#)

After you have created the SAP S/4HANA Cloud Extensibility service instance, either in the Cloud Foundry or Kyma environment, you need to configure event topics for the channel inside the SAP S/4HANA Cloud tenant, and then you need to create an SAP Event Mesh service instance, respectively in the Cloud Foundry or Kyma environment, for the application to consume SAP S/4HANA Cloud events. You do this in a dedicated JSON file that you add either when creating the service instance, or after that. To construct this JSON file, see [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#).

### Process Flow

Process Step	Related Documentation
1. Create an SAP S/4HANA Cloud Extensibility service instance with service plan messaging.	<ul style="list-style-type: none"><li>• <a href="#">Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment [page 2076]</a></li><li>• <a href="#">Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment [page 2078]</a></li></ul>
2. Define a dedicated JSON file that you add either when creating the SAP S/4HANA Cloud Extensibility service instance, or after that.	<a href="#">Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File [page 2080]</a>
3. Configure event topics for the channel inside SAP S/4HANA Cloud tenant.	<a href="#">Configure Event Topics in SAP S/4HANA Cloud [page 2088]</a>

## 5.4.3.3.2.1 Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment

Use this procedure to configure the communication between SAP S/4HANA Cloud and SAP Event Mesh.

### Prerequisites

- Before creating an SAP S/4HANA Cloud Extensibility service instance in the Cloud Foundry environment, see [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 2050\]](#).
- Have enabled Cloud Foundry environment for your subaccount. See [Enable Environment or Create Environment Instance \[page 2179\]](#).
- Have registered an SAP S/4HANA Cloud system. See [Register an SAP S/4HANA Cloud System in a Global Account in SAP BTP \[page 2002\]](#).
- Have configured the entitlements to the SAP S/4HANA Cloud Extensibility service. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

### Context

To configure the connectivity between the SAP S/4HANA Cloud tenant and Event Mesh and to enable the exchange of credentials between the two systems, you first need to create an [SAP S/4HANA Cloud Extensibility](#) service instance with service plan messaging. For more information about the messaging service plan, see [Supported Service Plans for SAP S/4HANA Cloud \[page 2101\]](#).

When creating this service instance, you create the required configurations in both the SAP S/4HANA Cloud tenant and the Event Mesh system associated with the subaccount in SAP BTP, so that events can flow.

### Procedure

1. In the SAP BTP cockpit, navigate to the subaccount in which you want to create a service instance.
2. In the navigation area, choose [Services](#) [Service Marketplace](#).  
All services available to you appear.
3. To enable the integration with an SAP S/4HANA Cloud system that you have registered in the global account in SAP BTP, choose [SAP S/4HANA Cloud Extensibility](#).
4. In the [SAP S/4HANA Cloud Extensibility](#) page, choose [Create](#).
5. In the [New Instance or Subscription](#) wizard:
  - a. In the [Service](#) dropdown list, ensure you have selected the [SAP S/4HANA Cloud Extensibility](#) service.
  - b. In the [Plan](#) dropdown list, select the [messaging](#) service plan.
  - c. In the [Runtime Environment](#) dropdown list, select [Cloud Foundry](#).
  - d. In the [Space](#) dropdown list, select your space. If you haven't create a space yet, you can do it at this point.

- e. In the *System Name* dropdown list, select the SAP S/4HANA Cloud system you have registered.
- f. In the *Instance Name* field, enter a name for your instance. Choose *Next*.
- g. Specify a JSON file or specify parameters in the JSON format to define the communication arrangement for the communication scenario *Enterprise Eventing Integration (SAP\_COM\_0092)* in the SAP S/4HANA Cloud tenant and to configure the parameters for the Enterprise Messaging service. If you decide to define the communication arrangement later on, you have to delete this service instance and create it again. Choose *Next*.

For more information about the structure of the JSON file, see [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#).

- h. Choose *Create*.

The newly created instance appears in the list of instances in the *Instance and Subscriptions* page.

Alternatively, you can create the instance using cf CLI. To do so, execute the following command

```
cf create-service s4-hana-cloud messaging emsconnect -c '{"systemName": "<system_name>","communicationArrangement": {"communicationArrangementName": "<communication_arrangement_name>","attributes": [{"name": "Channel","value": "<channel_name>"}, {"name": "Description","value": "<short_description>"}, {"name": "Topic Space","value": "<topic_to_be_used_by_events>"}, {"name": "QoS","value": "<quality_of_service>"}, {"name": "Reconnect Attempts","value": "<number_of_reconnect_attempts>"}, {"name": "Reconnect wait time(sec)","value": "<time_before_trying_to_reconnect>"}]}, "ems": {"parameters": {"emname": "enterprise.messaging_client_name", "namespace": "enterprise-messaging_client_namespace": {"options": {<required_options>}, "rules": {<required_rules>}}}}
```

For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

### Note

You can use the cf CLI to troubleshoot if the creation of the service instance fails. To do that, use this command in the cf CLI:

```
cf service <service_instance_name>
```

## Next Steps

[Configure Event Topics in SAP S/4HANA Cloud \[page 2088\]](#)

## 5.4.3.3.2.2 Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment

Use this procedure to configure the communication between SAP S/4HANA Cloud and SAP Event Mesh in the Kyma environment.

### Prerequisites

In the SAP BTP cockpit, you have assigned the `messaging` SAP S/4HANA Cloud service plan to the subaccount you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for the SAP BTP Subaccount \[page 2049\]](#).

### Context

To configure the connectivity between an SAP S/4HANA Cloud tenant and Event Mesh and to enable the exchange of credentials between the two systems, you first need to create an SAP S/4HANA Cloud Extensibility service instance with `messaging` service plan in the Kyma dashboard. For more information about the `messaging` service plan, see [Supported Service Plans for SAP S/4HANA Cloud \[page 2101\]](#).

When creating this service instance, you create the required configurations in both the SAP S/4HANA Cloud tenant and the Event Mesh system associated with the subaccount in SAP BTP, so that the events can flow from one system to the other.

### Procedure

1. In the SAP BTP cockpit, navigate to the subaccount for which you want to create an SAP S/4HANA Cloud Extensibility service instance.
2. On the subaccount *Overview* page in the *Kyma Environment* section, open the Kyma dashboard.
3. Navigate to the `default` namespace from the drop-down list in the top navigation.
4. Choose   from the left-hand side navigation.
5. In the *Service Instances* page, choose *Create Service Instance* in the upper right-hand corner. A new dialog opens.
6. Choose the *Simple* tab and fill in the following fields:
  - Give a meaningful name of the new SAP S/4HANA Cloud Extensibility service instance.
  - In the *Offering Name* field, enter `s4-hana-cloud`, which is the technical name of the SAP S/4HANA Cloud Extensibility service.
  - In the *Plan Name* field, enter `messaging`.
7. Choose the *YAML* tab.

In `parameters`: in the `spec:` section, specify the parameters to define the communication arrangement for the communication scenario *Enterprise Eventing Integration (SAP\_COM\_0092)* in the SAP S/4HANA

Cloud tenant and to configure the parameters for the Event Mesh service. Make sure you include at least these two required parameters: `systemName` and `emclientId`. The parameters are listed in this YAML file: [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#).

8. Choose *Create*.

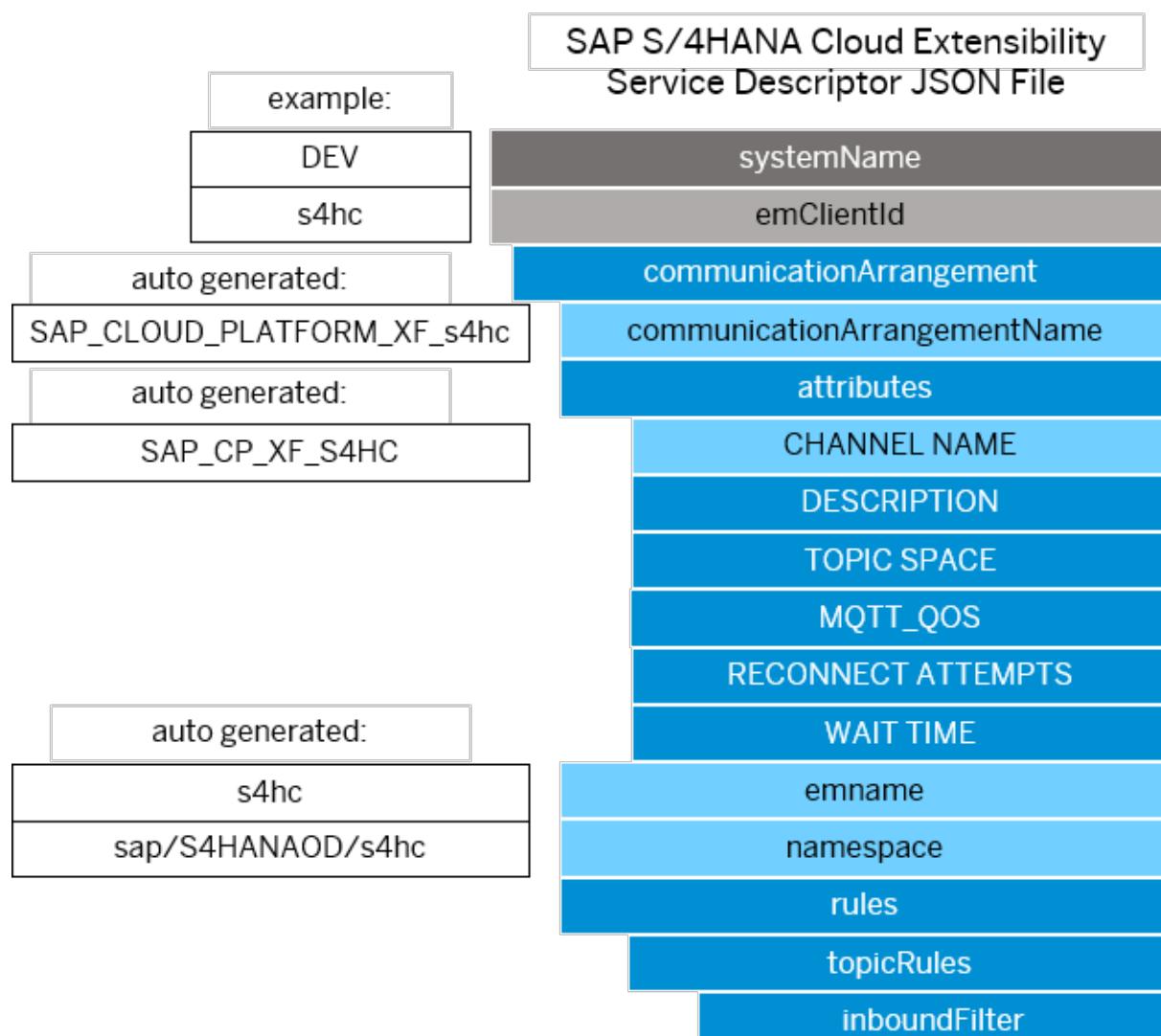
## Next Steps

After creating the SAP S/4HANA Cloud Extensibility service instance with the messaging plan, a respective messaging client is created for you in the `sap/S4HANAOD/{emClientId}` namespace of the Event Mesh service. As the next step, you must relate the two clients (SAP S/4HANA Cloud Extensibility and Event Mesh) in the Kyma dashboard. To do this, create an instance of the Event Mesh service with the details of the automatically created Event Mesh namespace. Follow the steps in [Create an SAP Event Mesh Service Instance in the Kyma Environment \[page 2091\]](#).

### 5.4.3.3.2.3 Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File

The SAP S/4HANA Cloud Extensibility service descriptor defines details of a message client and needs to be provided when provisioning new SAP S/4HANA Cloud Extensibility service instances with service plan messaging.

#### Procedure



#### Legend:

- Required property
- Required property whose value is used to generate automatically the values of a set of optional properties
- Optional property
- Optional property whose value is generated automatically based on the value of a required property. If specified, overrides the automatically generated one

Define the SAP S/4HANA Cloud Extensibility service descriptor in a JSON structure for the Cloud Foundry environment and in a YAML structure for the Kyma environment. It has to contain two sections:

- For the parameters needed to activate the communication arrangement for the communication scenario *Enterprise Eventing Integration (SAP\_COM\_0092)* in the SAP S/4 HANA Cloud tenant.  
There are only two required parameters, *emClientId* and *systemName*. The rest of the parameters are automatically generated. However, you can still provide the optional parameters in the descriptor to override the automatically generated values.
- For the parameters for configuring the SAP Event Mesh service.  
If this section is not explicitly added to the JSON/YAML file, the values of its parameters are automatically generated based on the values of the *emClientId* and *systemName* parameters.

Parameters required to activate the communication arrangement in SAP S4/HANA Cloud tenant

Parameter	Description
systemName	<p>The name of the system you have registered in the global account in SAP BTP.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>Required: Yes</li><li>The name must be the same as the one you have used when registering the SAP S/4HANA Cloud system during the pairing process.</li></ul>
emClientId	<p>Using default patterns, it generates <b>communication arrangement name</b>, <b>channel name</b>, <b>emname</b>, and <b>namespace</b>, when these parameters are not explicitly provided. The default values are described in the table below containing the parameters required to configure SAP Event Mesh service.</p> <p><b>① Note</b></p> <p>The values of the emname and namespace parameters must be the same as the values of the emname and namespace parameters in the SAP Event Mesh service descriptor. See <a href="#">Define SAP Event Mesh Service Descriptor JSON/YAML File [page 2093]</a>.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>Required: Yes</li><li>Allowed characters: <b>[a-zA-Z0-9 ]</b></li><li>Maximum length: 4</li></ul>
communicationArrangement	Defines the communication arrangement for the SAP S4/HANA Cloud tenant.

Parameter	Description
communicationArrangementName	<p>A communicationArrangement property.</p> <p>The name of the communication arrangement for the SAP S/4HANA Cloud tenant.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Allowed characters: <b>[a-zA-Z0-9_-]</b></li> <li>Maximum length: 80</li> <li>Default value: <b>"SAP_CLOUD_PLATFORM_XF_&lt;emClientId&gt;"</b></li> </ul>
attributes	<p>A communicationArrangement property.</p> <p>Defines the configuration properties for the communication arrangement.</p> <p>The name of the property is equivalent to the <i>Technical Property Name</i> column in the properties table in the Display Communication Scenarios app in the SAP S/4HANA Cloud system. See <a href="#">Display Communication Scenarios</a>.</p>
CHANNEL_NAME	<p>An attributes property.</p> <p>The name of the communication channel.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Allowed characters: <b>[A-Z0-9_-]</b></li> <li>Default value: <b>"SAP_CP_XF_&lt;emclientid&gt;"</b></li> </ul> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>Have in mind that the <b>&lt;emclientid&gt;</b> will be automatically capitalized.</p> </div> <ul style="list-style-type: none"> <li>Must be unique within the SAP S/4 HANA Cloud tenant.</li> </ul>

Parameter	Description
DESCRIPTION	<p>An <code>attributes</code> property.</p> <p>Short description.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Maximum length: 60</li> <li>Allowed characters: <code>[a-zA-Z0-9_-]</code></li> <li>Default value: <code>"Integration with Enterprise Messaging for EM Client: &lt;emclientID&gt;"</code></li> </ul>
TOPIC SPACE	<p>An <code>attributes</code> property.</p> <p>The identifier for the events that originate from the same source. This is the topic that the events should use.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Maximum length: 24</li> <li>Allowed characters: <code>[a-zA-Z0-9//]</code></li> <li>Contains exactly three segments, for example <code>a/b/c</code>.</li> <li>Default value: <code>"sap/S4HANAOD/&lt;emclientID&gt;"</code></li> <li>The <code>subscribeFilter</code> in the SAP Event Mesh configuration must be use the topic space as a prefix.</li> </ul>
MQTT_QOS	<p>An <code>attributes</code> property.</p> <p>Defines the quality of service.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Allowed values: <code>0, 1</code> <ul style="list-style-type: none"> <li>QoS=0, at most once delivery. The message is delivered according to the capabilities of the underlying network. No response is sent by the receiver and no retry is performed by the sender. The message arrives at SAP Event Mesh either once or not at all.</li> <li>QoS=1, at least once delivery. This quality of service ensures that the message arrives at SAP Event Mesh at least once.</li> </ul> </li> <li>Default value: <code>1</code></li> </ul>

Parameter	Description
RECONNECT_ATTEMPTS	<p>An <code>attributes</code> property.</p> <p>The number of attempts the Enterprise Event Enablement framework tries to reestablish the connection if the connection is lost.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Default value: <b>0</b></li> <li>If no value is entered or the entered value is 0, the framework tries to connect until connection is established. If the connection is not established after reaching reconnect attempts, the communication arrangement and the underlying channel is deactivated.</li> </ul>
WAIT_TIME	<p>An <code>attributes</code> property.</p> <p>Specifies the time (in seconds) for which the Enterprise Event Enablement framework waits before trying to reconnect. If the attempts fail, framework increases the wait time until the reconnect wait time (1800 seconds) is reached.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Default value: <b>10</b></li> </ul>

Parameters required to configure SAP Event Mesh service

Parameter	Description
emname	<p>Specifies the name of the SAP Event Mesh client. It is used by SAP Event Mesh to identify clients.</p> <p><b>Rules/Guidelines:</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Allowed characters: <b>[a-zA-Z0-9_-]</b></li> <li>Maximum length: 100</li> <li>Default value: <b>&lt;emClientId&gt;</b></li> <li>It is unique within a subaccount.</li> </ul>

Parameter	Description
namespace	<p>Namespace for the message client.</p> <p><b>Rules/Guidelines:</b> The namespace in the SAP Event Mesh configuration must be the same as the first three segments of the topic space in the configuration of the communication arrangement in SAP S4/HANA Cloud tenant.</p> <p><b>Rules/Guidelines:</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> <li>• Allowed characters: <code>[a-zA-Z0-9// ]</code></li> <li>• Maximum length: 24</li> <li>• It is unique within a subaccount</li> <li>• Contains exactly three segments, for example <code>a/b/c</code>.</li> <li>• Default value: <code>"sap/S4HANAOD/&lt;emclientid&gt;"</code>.</li> </ul>
rules	<p>Defines the access privileges of the message client.</p> <p><b>Guidelines/Rules:</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> <li>• In order to allow access to a queue or a topic the namespace of the corresponding owner client has to be added. The placeholder <code> \${namespace}</code> can be used instead of the defined namespace.</li> </ul>
topicRules	<p>A rules attribute.</p> <p><b>Guidelines/Rules:</b></p> <ul style="list-style-type: none"> <li>• Type: object</li> </ul>
inboundFilter	<p>A topicRules attribute.</p> <p>Defines if a client (publisher/producer) is allowed to send messages to the defined topics.</p> <p><b>Guidelines/Rules:</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> <li>• Allowed characters: <code>[a-zA-Z#?/ ]</code></li> <li>• Type: array</li> <li>• Default value: <code> \${namespace}/#</code></li> <li>• Example values: <code> \${namespace}/foo/bar, \${namespace}/#</code>.</li> </ul>

Parameter	Description
resources	<p>A collection of messaging resources such as queues, connections, and so on, required for a message client.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>
units	<p>A <code>resources</code> attribute.</p> <p>Specify this value in the service descriptor to allocate messaging resources based a specific business scenario. See <a href="#">Syntax for Service Descriptor</a>.</p> <p><b>Guidelines/Rules:</b></p> <ul style="list-style-type: none"> <li>Required: No</li> <li>Allowed values: <b>10-50</b></li> <li>Type: attribute</li> <li>Default value: <b>10</b></li> </ul>

## JSON File Example

In this example, only the required properties are provided.

```
{
  "systemName": "DEV",
  "emClientId": "s4hc"
}
```

This descriptor is equivalent to:

### ⓘ Note

All properties can be overridden explicitly.

```
{
  "systemName": "DEV",
  "emClientId": "s4hc",
  "communicationArrangement": {
    "communicationArrangementName": "SAP_CLOUD_PLATFORM_XF_s4hc",
    "attributes": [
      {
        "name": "CHANNEL_NAME",
        "value": "SAP_CP_XF_S4HC"
      },
      {
        "name": "DESCRIPTION",
        "value": "Integration with Enterprise Messaging for EM Client: s4hc"
      },
      {
        "name": "TOPIC_SPACE",
        "value": "sap/S4HANAOD/s4hc"
      }
    ]
}
```

```

        {
            "name": "MQTT_QOS",
            "value": "1"
        },
        {
            "name": "RECONNECT ATTEMPTS",
            "value": "0"
        },
        {
            "name": "WAIT TIME",
            "value": "10"
        }
    ]
},
"ems": {
    "parameters": {
        "emname": "s4hc",
        "namespace": "sap/S4HANAOD/s4hc",
        "rules": {
            "topicRules": {
                "inboundFilter": [
                    "${namespace}/#"
                ]
            }
        },
        "resources": {
            "units": "10"
        }
    }
}
}

```

## YAML File Example

In this example, only the required properties are provided.

```

parameters:
  systemName: DEV
  emClientId: s4hc

```

This descriptor is equivalent to:

### ⓘ Note

All properties can be overridden explicitly.

```

spec:
  externalName: ''
  serviceOfferingName: s4-hana-cloud
  servicePlanName: messaging
parameters:
  systemName: DEV
  emClientId: s4hc
  communicationArrangement:
    communicationArrangementName: SAP_CLOUD_PLATFORM_XF_s4hc
    attributes:
      - name: CHANNEL NAME
        value: SAP_CP_XF_S4HC
      - name: DESCRIPTION

```

```

        value: 'Integration with Enterprise Messaging for EM Client: s4hc'
- name: TOPIC SPACE
  value: sap/S4HANAOD/s4hc
- name: MQTT_QOS
  value: '1'
- name: RECONNECT ATTEMPTS
  value: '0'
- name: WAIT TIME
  value: '10'
ems:
  parameters:
    emname: s4hc
    namespace: sap/S4HANAOD/s4hc
    rules:
      topicRules:
        inboundFilter:
        - "${namespace}/#"
  resources:
    units: '10'

```

#### 5.4.3.3.2.4 Configure Event Topics in SAP S/4HANA Cloud

Configure event topics for the channel inside the SAP S/4HANA Cloud tenant.

#### Prerequisites

You have created the SAP S/4HANA Cloud Extensibility service instance in the Cloud Foundry or Kyma environment. See:

- [Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment \[page 2076\]](#)
- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2078\]](#)

#### Context

After you have created the SAP S/4HANA Cloud Extensibility service instance, you need to configure event topics for the channel inside the SAP S/4HANA Cloud tenant, and then you need to create an SAP Event Mesh service instance for the application to consume SAP S/4HANA Cloud events.

#### Procedure

Configure event topics for the channel inside SAP S/4HANA Cloud tenant. Use the channel you have specified in SAP S/4HANA Cloud Extensibility service descriptor JSON file when configuring the parameters for the communication arrangement in SAP S4/HANA Cloud tenant.

As an example, you can use the [\*sap/s4/beh/businesspartner/v1/BusinessPartner/Changed/v1\*](#) outbound topic.

## Next Steps

Create an SAP Event Mesh service instance in the Cloud Foundry or the Kyma environment. See:

- [Create an SAP Event Mesh Service Instance in the Cloud Foundry Environment \[page 2090\]](#)
- [Create an SAP Event Mesh Service Instance in the Kyma Environment \[page 2091\]](#)

### 5.4.3.3.3 Enable the SAP Event Mesh Service for Your Subaccount in SAP BTP

#### Prerequisites

- You have created the SAP S/4HANA Cloud Extensibility service instance in the Cloud Foundry or Kyma environment. See:
  - [Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment \[page 2076\]](#)
  - [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2078\]](#)
- You have configured event topics for the channel inside SAP S/4HANA Cloud tenant. See [Configure Event Topics in SAP S/4HANA Cloud \[page 2088\]](#).

#### Overview

To enable the SAP Event Mesh for your subaccount in SAP BTP, you have to create an instance of the SAP Event Mesh service with service plan `default`. This instance will specify the details of the namespace that you created in SAP Event Mesh for the SAP S/4HANA Cloud Extensibility service instance with the messaging plan. This way, you are setting up a contract between SAP Event Mesh and the Cloud Foundry, or the Kyma environment, and connect them to enable seamless event flow from the given SAP S/4HANA Cloud tenant.

Depending on the SAP BTP environment, to create an SAP Event Mesh service instance, you have to choose one of these:

- [Create an SAP Event Mesh Service Instance in the Cloud Foundry Environment \[page 2090\]](#)
- [Create an SAP Event Mesh Service Instance in the Kyma Environment \[page 2091\]](#)

Define the details of your message client in SAP Event Mesh as parameters in a dedicated JSON file. To map the message client to the client defined in the SAP S/4HANA Cloud JSON file, in the SAP Event Mesh JSON file, you must include the `subscribeFilter` parameter which specifies the `sap/S4HANAOD/{emClientId}` namespace you created for the SAP S/4 HANA Cloud Extensibility instance of the messaging plan. For more information about the structure of the JSON file, see [Define SAP Event Mesh Service Descriptor JSON/YAML File \[page 2093\]](#).

## Process Flow

Process Step	Related Documentation
1. Create an SAP Event Mesh service instance with service plan <code>default</code> .	<ul style="list-style-type: none"><li>• <a href="#">Create an SAP Event Mesh Service Instance in the Cloud Foundry Environment [page 2090]</a></li><li>• <a href="#">Create an SAP Event Mesh Service Instance in the Kyma Environment [page 2091]</a></li></ul>
2. Define a dedicated JSON file that you add either when creating the SAP Event Mesh service instance, or after that.	<a href="#">Define SAP Event Mesh Service Descriptor JSON/YAML File [page 2093]</a>
3. Create a queue in the message client that refers to the SAP Event Mesh service instance and then subscribe the topic of the namespace of the SAP S/4HANA Cloud Extensibility service to that queue.	<a href="#">Create Queues and Subscribe to Them [page 2099]</a>

### 5.4.3.3.1 Create an SAP Event Mesh Service Instance in the Cloud Foundry Environment

Use this procedure to enable the SAP Event Mesh service for the subaccount where your extension application will reside.

#### Context

To enable the SAP Event Mesh service for your subaccount in SAP BTP, you have to go through the following steps.

#### Procedure

1. Prepare a JSON file that contains details of a message client. See [Define SAP Event Mesh Service Descriptor JSON/YAML File \[page 2093\]](#).
2. Create an SAP Event Mesh service instance for the application to consume SAP S/4HANA Cloud events.
  - a. In the SAP BTP cockpit, navigate to the space in which you want to create a service instance.
  - b. In the navigation area, choose , and then choose *Event Mesh* in the *Service Marketplace* panel.
  - c. From the *Event Mesh* service tile, choose *Create* and follow the steps in the wizard to subscribe to the service.
  - d. In the *New Instance or Subscription* wizard:

1. In the *Service* dropdown list, select *Event Mesh*.
2. In the *Plan* dropdown list select the *default* service plan.
3. In the *Runtime Environment* dropdown list, select *Cloud Foundry*.
4. In the *Space* dropdown list, select your Cloud Foundry space.
5. In the *Instance Name* field, enter a name for your instance. Choose *Next*.
6. Specify the JSON file you prepared in **Step 1**. Choose *Next*.
7. Review and verify the instance details, and choose *Create*.

## Next Steps

[Create Queues and Subscribe to Them \[page 2099\]](#)

### 5.4.3.3.3.2 Create an SAP Event Mesh Service Instance in the Kyma Environment

Use this procedure to enable the SAP Event Mesh service for the subaccount where your extension application will reside.

## Prerequisites

- In the SAP BTP cockpit, you have assigned the messaging SAP S/4HANA Cloud Extensibility service plan to the SAP BTP subaccount that you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).
- In the Kyma dashboard, you have created the SAP S/4HANA Cloud Extensibility service instance with the messaging plan. See [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2078\]](#).
- In the SAP BTP cockpit, you have assigned the *default* SAP Event Mesh service plan to the SAP BTP subaccount that you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

## Context

In the Kyma dashboard, create an instance of the SAP Event Mesh service. This instance will specify the details of the namespace that you created in SAP Event Mesh for the SAP S/4HANA Cloud Extensibility service instance with the *default* plan. This way, you are setting up a contract between SAP Event Mesh and the Kyma environment, and connect them to enable seamless event flow from the given SAP S/4HANA Cloud tenant to Kyma.

## Procedure

1. In the SAP BTP cockpit, navigate to the subaccount for which you want to create an SAP Event Mesh service instance.
  2. On the subaccount *Overview* page in the *Kyma Environment* section, open the Kyma dashboard.
  3. Navigate to the *default* namespace from the drop-down list in the top navigation.
  4. Choose  *Service Management*  *BTP Service Instances*  from the left-hand side navigation.
  5. In the *Service Instances* page, choose *Create Service Instance* in the upper right-hand corner. A new dialog opens.
  6. Choose the *Simple* tab and fill in the following fields:
    - Give a meaningful name of the new SAP Event Mesh service instance.
    - In the *Offering Name* field, enter **enterprise-messaging**, which is the technical name of the SAP Event Mesh service.
    - In the *Plan Name* field, enter **default**.
  7. Choose the *YAML* tab.
- In *parameters*: in the *spec:* section, specify the parameters to define the details of your message client in the SAP Event Mesh service. You must include the `subscribeFilter` parameter which specifies the `sap/S4HANAOD/{emClientId}` namespace you created for the SAP S/4 HANA Cloud Extensibility instance of the messaging plan in SAP Event Mesh. For more information about the structure of the YAML file, see [Define SAP Event Mesh Service Descriptor JSON/YAML File \[page 2093\]](#).
8. Choose *Create*.

## Results

Once the service instance is provisioned, you can see it under   *Service Management*  *Instances*  in the left-hand side navigation of the Kyma dashboard.

## Next Steps

[Create Queues and Subscribe to Them \[page 2099\]](#)

### 5.4.3.3.3 Define SAP Event Mesh Service Descriptor JSON/YAML File

The SAP Event Mesh service descriptor defines details of a message client and needs to be provided when provisioning new SAP Event Mesh service instances with service plan default.

#### Prerequisites

- You have created an SAP S/4HANA Cloud Extensibility service instance with defined SAP S/4HANA Cloud Extensibility descriptor. See [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#).
- Get to know the additional parameters you can have in the SAP Event Mesh service descriptor. In this page only the parameters that are related to this use case are described. See [Syntax for Service Descriptor in SAP Event Mesh documentation](#).

#### Context

You set up service instance of the SAP Event Mesh service to receive the events which are sent by the SAP S/4HANA Cloud Extensibility service. In the JSON file for the Cloud Foundry environment and in the YAML file for the Kyma environment, this service instance needs to be created with a different emname as well as a different namespace. The rule set here is important as each client should only publish in its own namespace. Since this message client's purpose is to receive messages from a different client, the rules of the subscribeFilter for topics need to take care of that. So the value of the message client defined in the SAP Event Mesh JSON/YAML file has to be configured to have the value of the subscribeFilter to be the same as the value of the namespace of the other client defined in the SAP S/4HANA Cloud Extensibility JSON/YAML file.

After both message clients are set up, you have to create a queue in the message client which refers to the SAP Event Mesh service instance. You can name the queue as you like and it will be created in the namespace of the client. This queue can now be subscribed to the topic configured **step 7** in the [Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment \[page 2076\]](#) page. This topic is defined in the namespace of the S/4HANA Cloud Extensibility service descriptor, for example **sap/S4HANAOD/s4hc/myTopic**.

The service instance of SAP Event Mesh is the message client which consumes the messages and should be bound to the actual application. These are the important aspects of the whole configuration:

- When creating a service instance of the SAP S/4HANA Cloud Extensibility service:
  - Use unique emname per subaccount
  - Use unique namespace per subaccount, for example **a/b/c**
- When creating a service instance of the SAP Event Mesh service:
  - Use unique emname per subaccount
  - Use unique namespace per subaccount, for example **d/e/f**
  - Use the namespace defined in the SAP S/4HANA Cloud Extensibility JSON/YAML file and put it in the subscribeFilter parameter

JSON:

#### ↔ Sample Code

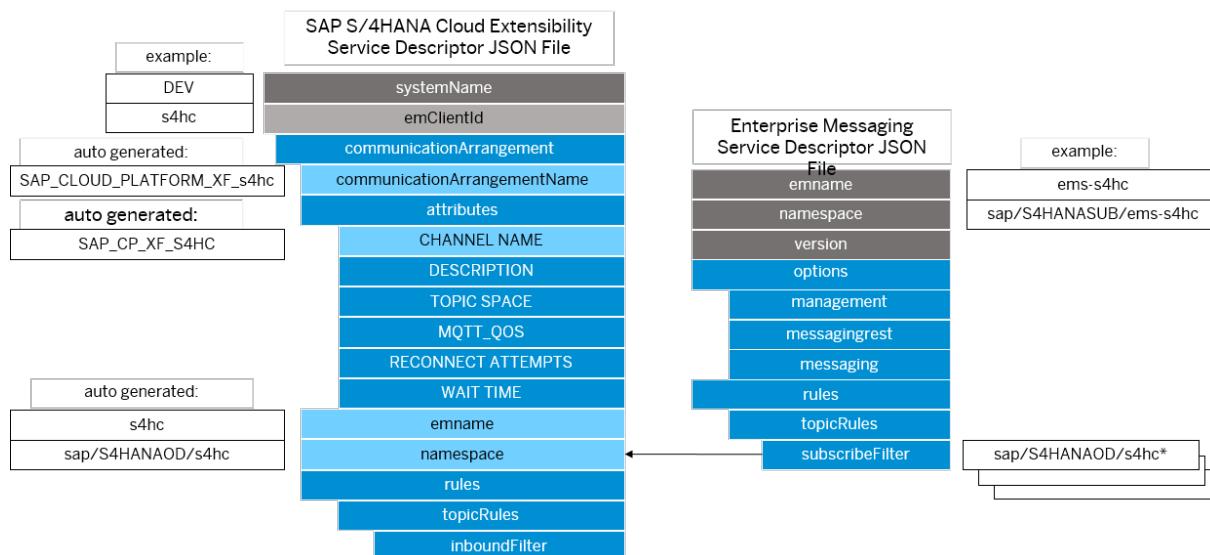
```
"subscribeFilter": [  
  "a/b/c/*",  
  "${namespace}/*"  
]
```

YAML:

#### ↔ Sample Code

```
subscribeFilter:  
- a/b/c/*  
- "${namespace}/*"
```

- Create queue for this client
- Subscribe this queue to the topic defined in the namespace of the SAP S/4HANA Cloud Extensibility JSON/YAML file



Legend:

	Required property
	Required property whose value is used to generate automatically the values of a set of optional properties
	Optional property
	Optional property whose value is generated automatically based on the value of a required property. If specified, overrides the automatically generated one

## Procedure

Define the JSON file for the SAP Event Mesh service instance following the requirements for the properties and their values listed below:

Parameter	Description
emname	<p>Specifies the name of the SAP Event Mesh client. It is used by SAP Event Mesh to identify clients.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"><p><b> ⓘ Note</b></p><p>The emname parameter in this descriptor must be unique per subaccount, so it must have the different value than the emname parameter in the SAP S/4HANA Cloud service descriptor. See <a href="#">Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File [page 2080]</a>.</p></div> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>• Required: Yes</li><li>• Allowed characters: [a-zA-Z0-9_-]</li><li>• Max length: 100</li></ul> <p><b>Rules/Guidelines:</b> It is unique within a subaccount.</p>
version	<p>It specifies the version of the service descriptor.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"><p><b> ⓘ Note</b></p><p>The version parameter is optional when you use the deprecated syntax. See <a href="#">Syntax for Service Descriptor</a>.</p></div> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>• Required: Yes</li><li>• Allowed value: 1.1.0</li></ul>

Parameter	Description
namespace	<p>Namespace for the messaging client.</p> <p><b> ⓘ Note</b></p> <p>The <code>namespace</code> parameter in this descriptor must be unique per subaccount, so it must have the different value than the <code>namespace</code> parameter in the SAP S/4HANA Cloud service descriptor. See <a href="#">Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File [page 2080]</a>.</p>
	<p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes</li> <li>• Allowed characters: [a-zA-Z0-9//]</li> <li>• Max length: 24</li> <li>• It is unique within a subaccount</li> <li>• Contains exactly three segments, for example <b>d/e/f</b>.</li> </ul>
options	Defines the access channels for the message client.
management	<p>An <code>options</code> attribute.</p> <p>Enables/disables the usage of the management REST APIs.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Default: false</li> <li>• Allowed values: true   false</li> </ul>
• messagingrest	<p>An <code>options</code> attribute.</p> <p>Enables/Disables the usage of the messaging REST APIs.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Default: false</li> <li>• Allowed values: true   false</li> </ul>
messaging	<p>An <code>options</code> attribute.</p> <p>Enables/disables the usage of the messaging gateway.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Default: true</li> <li>• Allowed values: true   false</li> </ul>

Parameter	Description
rules	<p>Defines the access privileges of the messaging client.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: true</li> <li><b>Guidelines/Rules:</b> In order to allow access to a queue or a topic, the namespace of the corresponding owner client has to be added. The placeholder \${namespace} can be used instead of the defined namespace.</li> </ul>
topicRules	<p>A rules attribute.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Type: Object</li> </ul>
subscribeFilter	<p>A topicRules attribute.</p> <p>Filters from which topics a client (receiver) is allowed to receive messages.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Type: Array</li> <li>Allowed characters: [a-zA-Z*?/]</li> <li>Default: no default</li> <li>Example values: \${namespace}/foo/bar/*, \${namespace}/*.</li> </ul>

**Note**

The namespace part of the value must be specified exactly as the namespace value of the SAP S/4HANA Cloud Extensibility service descriptor JSON File described in [Define SAP S/4HANA Cloud Extensibility Service Descriptor JSON/YAML File \[page 2080\]](#).

## JSON File Example

```
{
  "emname": "ems-s4hc",
  "namespace": "sap/S4HANAOD/ems-s4hc",
  "version": "1.1.0",
  "options": {
    "management": true,
    "messagingrest": true,
    "messaging": true
  },
  "rules": {
    "topicRules": [
      {
        "subscribeFilter": [
          {
            "topic": "foo/bar/*"
          }
        ],
        "rules": [
          {
            "rule": "${namespace}/foo/bar/*"
          }
        ]
      }
    ]
  }
}
```

```

        "queueRules": {
            "subscribeFilter": [
                "${namespace}/*"
            ]
        },
        "topicRules": {
            "subscribeFilter": [
                "${namespace}/*",
                "sap/S4HANAOD/s4hc/*"
            ]
        }
    }
}

```

## YAML File Example

```

spec:
  externalName: ''
  serviceOfferingName: enterprise-messaging
  servicePlanName: default
parameters:
  emname: ems-s4hc
  namespace: sap/S4HANAOD/ems-s4hc
  version: 1.1.0
options:
  management: true
  messagingrest: true
  messaging: true
rules:
  queueRules:
    subscribeFilter:
      - "${namespace}/*"
  topicRules:
    subscribeFilter:
      - "${namespace}/*"
      - sap/S4HANAOD/s4hc/*

```

## Related Information

[Syntax for Service Descriptor in SAP Event Mesh documentation](#)

## 5.4.3.3.4 Create Queues and Subscribe to Them

### Prerequisites

You have created the SAP S/4HANA Cloud Extensibility service instance in the Cloud Foundry or Kyma environment. See:

- [Create an SAP S/4HANA Extensibility Service Instance in the Cloud Foundry Environment \[page 2076\]](#)
- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 2078\]](#)

You have created the SAP Event Mesh service instance in the Cloud Foundry or Kyma environment. See:

- [Create an SAP Event Mesh Service Instance in the Cloud Foundry Environment \[page 2090\]](#)
- [Create an SAP Event Mesh Service Instance in the Kyma Environment \[page 2091\]](#)

### Context

After both instances are created and configured, you can create the topic-to-queue subscription. To do so, you first create a subscription to SAP Event Mesh with plan standard. Then, create a queue in the client that refers to the SAP Event Mesh service instance and then subscribe the topic of the namespace of the SAP S/4HANA Cloud Extensibility service to that queue.

#### ⓘ Note

You can subscribe to the SAP Event Mesh service once per subaccount.

### Procedure

1. Create a subscription to SAP Event Mesh.
  - a. In the SAP BTP cockpit, navigate to the space in which you want to create a service instance.
  - b. In the navigation area, choose **Services** **Service Marketplace**, and then choose **Event Mesh** in the **Service Marketplace** panel.
  - c. From the **Event Mesh** service tile, choose **Create** and follow the steps in the wizard to subscribe to the service.
  - d. In the **New Instance or Subscription** wizard:
    1. In the **Service** dropdown list, select **Event Mesh**.
    2. In the **Plan** dropdown list, select the **standard** service plan. Choose **Create**.
2. Assign the necessary roles to your user.
  - a. In the SAP BTP cockpit, navigate to your subaccount.

- b. In the navigation area, choose [Security](#) [Trust Configuration](#) , and then choose [SAP ID Service](#) in the [Trust Configuration](#) panel.
  - c. On the [Role Collection Assignment](#) page, enter the e-mail of your user in the empty field and choose [Show Assignments](#). Then, choose [Assign Role Collection](#).
  - d. In the [Assign Role Collection](#) wizard, select the following role collections:
    - [Enterprise Messaging Administrator](#)
    - [Enterprise Messaging Developer](#)
    - [Enterprise Messaging Display](#)
    - [Enterprise Messaging Subscription Administrator](#)
  - e. Choose [Assign Role Collection](#).
3. Open the SAP Event Mesh application.
    - a. In the SAP BTP cockpit, navigate to the space in which you want to create a service instance.
    - b. In the navigation area, choose [Services](#) [Instances and Subscriptions](#) .
    - c. In the [Subscriptions](#) tab, choose the [Event Mesh](#) application.
    - d. From the [Event Mesh](#) page, choose [Go to Application](#) and open the SAP Event Mesh application.

**Note**

If you are using the sample JSON files when creating the SAP S/4HANA Extensibility and the Event Mesh service instances, you should see 2 message clients in the SAP Event Mesh application, [s4hc](#) and [ems-s4hc](#).

4. Create a queue that is specific to your application.
  - a. Open the message client created when creating the Event Mesh service instance. For example, choose the [ems-s4hc](#) tile.
  - b. Choose the [Queues](#) tab in the message client page and then choose [Create Queue](#).
  - c. Enter a queue name. Note that in the blue field of the [Create a New Queue](#) wizard, the final queue name is displayed. For example, if you write [my-queue](#) in the [Queue Name](#) field, the final queue name would be [sap/S4HANAOD/ems-s4hc/my-queue](#). Choose [Create](#).
5. Subscribe this queue to the channel topic that SAP S/4HANA Cloud tenant uses to produce events.
  - a. From the [Actions](#) of your newly created queue, select [Queue Subscriptions](#).
  - b. In the [Topic](#) field, enter the topic you configured in the SAP S/4HANA Cloud tenant. For example, [sap/S4HANAOD/s4hc/ce/sap/s4/beh/businesspartner/v1/BusinessPartner/Changed/v1](#).
  - c. Choose [Add](#).

## Related Information

[Manage Queues](#)

[REST APIs for Events](#)

## 5.4.3.4 Supported Service Plans for SAP S/4HANA Cloud

The following service plans are available for subaccounts in SAP BTP paired with an SAP S4/HANA Cloud tenant.

### api-access

This service plan enables all communication scenarios, both predefined and custom, which allow you to consume SAP S/4HANA Cloud APIs and integrate your extension applications with the respective SAP S/4HANA Cloud functionality.

- To access the specific documentation of these APIs, see [SAP S/4HANA Cloud APIs at SAP Business Accelerator Hub](#).
- For more information about SAP S/4HANA Cloud communication scenarios see:
  - [Communication Management](#)
  - [Custom Communication Scenarios](#)

#### ⓘ Note

These service plans have been deprecated:

- [sap\\_com\\_0109](#)
- [sap\\_com\\_0009](#)
- [sap\\_com\\_0008](#)

However, you can still enable these communication scenarios using the [api-access](#) service plan:

- *Sales Order Integration (SAP\_COM\_0109)*: allows you to integrate your extension applications with sales order processing in SAP S/4HANA Cloud. For more information, see [https://api.sap.com/api/API\\_SALES\\_ORDER\\_SRV/overview](https://api.sap.com/api/API_SALES_ORDER_SRV/overview).
- *Product Integration (SAP\_COM\_0009)*: enables you to replicate product master data from client system to SAP S/4HANA system. For more information, see <https://api.sap.com/api/PRODUCTMDMBULKREPLICAREQUEST/overview>.
- *Business Partner, Customer and Supplier Integration (SAP\_COM\_0008)*: allows you to consume the Business Partner API which enables you to create, read, update, and delete master data related to Business Partners, Suppliers, and Customers in an SAP S/4HANA system. For more information, see [https://api.sap.com/api/API\\_BUSINESS\\_PARTNER/overview](https://api.sap.com/api/API_BUSINESS_PARTNER/overview).

For a sample JSON for these communication scenarios, see [Communication Arrangement JSON/YAML File - Properties \[page 2056\]](#).

### messaging

This service plan enables the *Enterprise Eventing Integration (SAP\_COM\_0092)* communication scenario which allows you to consume SAP S/4HANA Cloud events and create event-based extensions.

#### Note

This service plan has been renamed from [sap\\_com\\_0092](#).

### 5.4.3.5 Configuring Backup

You can back up and restore the destination and the trust configuration settings for the integration between the SAP S/4HANA Cloud and the global account in SAP BTP.

You can back up and restore the following configuration settings:

- The HTTP destination configured during the creation of the SAP S/4HANA Cloud Extensibility service instance. For more, see [Export Destinations](#).
- Your subaccount X.509 certificates. For more information, see [Set up Trust Between Systems](#)
- Your subaccount-specific IdP metadata.  
To download your SAP SuccessFactors IdP SAML 2.0 metadata, in the SAP BTP cockpit, navigate to  
 [Connectivity](#)  [Destinations](#)  [Download IdP Metadata](#) 
- Your identity provider configurations.  
You can back up identity provider configurations with the Identity Provider Management API. For more information see: [Identity Provider Management](#)  API.

### 5.4.3.6 Auditing and Logging Information

Here you can find a list of the events that are logged by SAP S/4HANA Cloud Extensibility service. To retrieve the audit logs stored for SAP S/4HANA Cloud Extensibility create a support ticket in component BC-NEO-EXT-S4C.

Events written in audit logs

Event grouping	What events are logged	How to identify related log events	Additional information
Creating an integration	Start creating integration	<ul style="list-style-type: none"> <li>• "type": "s4Integration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "new": &lt;integration_details&gt;} ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies an integration of a global account with an SAP S/4HANA Cloud system has been started. The log contains integration details such as the SAP S/4HANA Cloud tenant host name and ID, the Identity Authentication tenant ID, the integration ID, and other.</p> <p>The <code>customDetails</code> attribute contains the SAP S/4HANA Cloud user ID of the user who has triggered the integration process.</p>
Created integration		<ul style="list-style-type: none"> <li>• "type": "s4Integration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "new": &lt;integration_details&gt;} ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies an integration of a global account with an SAP S/4HANA Cloud system has been completed. The log contains integration details such as the SAP S/4HANA Cloud tenant host name and ID, the Identity Authentication tenant ID, the integration ID, and other.</p> <p>The <code>customDetails</code> attribute contains the SAP S/4HANA Cloud user ID of the user who has triggered the integration process.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	Delete integration	<ul style="list-style-type: none"> <li>• "type": "s4Integration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": &lt;integration_details&gt; } ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies a deletion of an integration of a global account with an SAP S/4HANA Cloud system has been triggered. The log contains integration details such as the SAP S/4HANA Cloud tenant host name and ID, the Identity Authentication tenant ID, the integration ID, and other.</p> <p>The <code>customDetails</code> attribute contains the SAP S/4HANA Cloud user ID of the user who has triggered the deletion process.</p>
	Deleted integration	<ul style="list-style-type: none"> <li>• "type": "s4Integration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": &lt;integration_details&gt; } ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies a deletion of an integration of a global account with an SAP S/4HANA Cloud system has been completed. The log contains integration details such as the SAP S/4HANA Cloud tenant host name and ID, the Identity Authentication tenant ID, the integration ID, and other.</p> <p>The <code>customDetails</code> attribute contains the SAP S/4HANA Cloud user ID of the user who has triggered the deletion process.</p>
Setting up the connectivity	Start creating connectivity configuration	<ul style="list-style-type: none"> <li>• "type": "s4Configuration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "new": &lt;connection_details&gt; } ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies a creation of a service instance of the SAP S/4HANA Cloud Extensibility service has been triggered. The log contains details such as the global account ID, the subaccount ID, the communication arrangement, and the authentication mechanism.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	Created connectivity configuration	<ul style="list-style-type: none"> <li>• "type": "s4Configuration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": &lt;connection_details&gt; } ]</li> <li>• "status": "BEGIN"</li> </ul>	Event that signifies a creation of a service instance of the SAP S/4HANA Cloud Extensibility service has been completed. The log contains details such as the global account ID, the subaccount ID, the communication arrangement, and the authentication mechanism.
	Start deleting connectivity configuration	<ul style="list-style-type: none"> <li>• "type": "s4Configuration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": &lt;connection_details&gt; } ]</li> <li>• "status": "BEGIN"</li> </ul>	Event that signifies the deletion of a service instance of the SAP S/4HANA Cloud Extensibility service has been started. The log contains details such as the global account ID, the subaccount ID, the communication arrangement, and the authentication mechanism.
	Deleted connectivity configuration	<ul style="list-style-type: none"> <li>• "type": "s4Configuration"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": &lt;connection_details&gt; } ]</li> <li>• "status": "END"</li> </ul>	Event that signifies the deletion of a service instance of the SAP S/4HANA Cloud Extensibility service has been completed. The log contains details such as the global account ID, the subaccount ID, the communication arrangement, and the authentication mechanism.

The following information is described in the table columns:

- *Event grouping* - Events that are logged with a similar format or are related to the same entities.
- *What events are logged* - Description of the security or data protection and privacy related event that is logged.
- *How to identify related log events* - Search criteria or key words, that are specific for a log event that is created along with the logged event.
- *Additional information* - Any related information that can be helpful.

## Related Information

[Audit Logging in the Cloud Foundry Environment](#)

### **5.4.3.7 Accessibility Features in SAP S/4HANA Cloud Extensibility Service**

To optimize your experience of SAP S/4HANA Cloud Extensibility service, SAP Business Technology Platform (SAP BTP) provides features and settings that help you use the software efficiently.

#### **ⓘ Note**

SAP S/4HANA Cloud Extensibility service runs on the SAP BTP cockpit. For this reason, accessibility features for SAP BTP cockpit also apply. See the accessibility documentation for SAP BTP cockpit on SAP Help Portal at [Accessibility Features in SAP BTP cockpit \[page 2141\]](#).

For more information on screen reader support and keyboard shortcuts, see [Accessibility for End Users](#).

### **5.4.3.8 Troubleshooting for SAP S/4HANA Cloud Extensibility Service**

Information about troubleshooting issues you might experience when extending SAP S/4HANA Cloud on SAP BTP using the SAP S/4HANA Cloud Extensibility service.

#### **Cloud Foundry Environment**

These are some issues you might experience when extending SAP S/4HANA Cloud on SAP BTP, Cloud Foundry environment:

- [Cannot create an SAP S/4HANA Cloud Extensibility Service Instance of the Messaging Plan \[page 2106\]](#)
- [My SAP S/4HANA Cloud Extensibility Service Instance Failed to Create and It Didn't Say Why \[page 2107\]](#)

#### **5.4.3.8.1 Cannot create an SAP S/4HANA Cloud Extensibility Service Instance of the Messaging Plan**

Cannot create an SAP S/4HANA Cloud Extensibility service instance of the messaging plan.

#### **Issue/Symptom**

You are trying to create an SAP S/4HANA Cloud Extensibility service instance of the messaging plan and you get the following error message:"Creation Failed."

## Reason

You are probably providing in the JSON file the same `emClientId` that was already used in the subaccount.

## Solution

Make the value of the `emClientId` parameter unique within the subaccount.

## Related Information

[Set Up the Connectivity Between Event Mesh and the SAP S/4HANA Cloud Tenant \[page 2075\]](#)

## 5.4.3.8.2 My SAP S/4HANA Cloud Extensibility Service Instance Failed to Create and It Didn't Say Why

My SAP S/4HANA Cloud Extensibility service instance failed to create and it didn't say why.

### Issue/Symptom

You are trying to create an SAP S/4HANA Cloud Extensibility service instance but the creation fails with no error message.

## Solution

This is what you can do:

- You can use the Cloud Foundry command line interface to get the error message by executing the command:  
`cf service <service_instance_name>`  
Check for any error messages there.
- Can you reach the SAP S/4HANA Cloud tenant? Check that the tenant is not in maintenance or being updated.

If this didn't help, you can try some more specific actions. What service plan are you using?

### I am using the messaging service plan

#### Issue/Symptom

Your SAP S/4HANA Cloud Extensibility service instance of the messaging plan failed to create and it didn't say why.

### Solution

Try to do the following:

- Is the Enterprise Messaging Integration scenario (SAP\_COM\_0092) enabled on your SAP S/4HANA Cloud tenant? Check whether you see it in the **Display Communication Scenarios** application in your SAP S/4HANA Cloud tenant.
- The clients for SAP Event Mesh require unique values (within your subaccount in SAP BTP) for the **emname** and **namespace** properties. In the *messaging* plan, you have either provided these manually, or you have only set the **emClientId** property and let it generate them. If the latter, then these values would be **emClientId>** for **emname** and **sap/S4HANAOD/emClientId>** for **namespace**. Check whether your subaccount already has existing clients with such **emname** or **namespace** values. You can do this by using the **Enterprise Messaging Business Application** from the SAP Event Mesh subscription of your subaccount. See [View Event Catalog for a Subaccount](#).

## I am using the api-access service plan

### Issue/Symptom

Your SAP S/4HANA Cloud Extensibility service instance of the api-access plan failed to create and it didn't say why.

### Solution

Try to do the following:

- Is the communication scenario with the **scenarioId** that you provided enabled on that tenant? Check whether you see it in the **Display Communication Scenarios** application in your SAP S/4HANA Cloud tenant.
- Are you using supported authentication types? In the scenario definition, you can see which authentication types are supported for both outbound and inbound connectivity. Verify that the types you are providing in the descriptor are supported by the scenario.

## 5.4.4 Extending SAP Marketing Cloud in the Cloud Foundry and Kyma Environment

### Overview

#### Note

The SAP S/4HANA Cloud Extensibility service is available for the EU-only access regions. See [Regions](#).

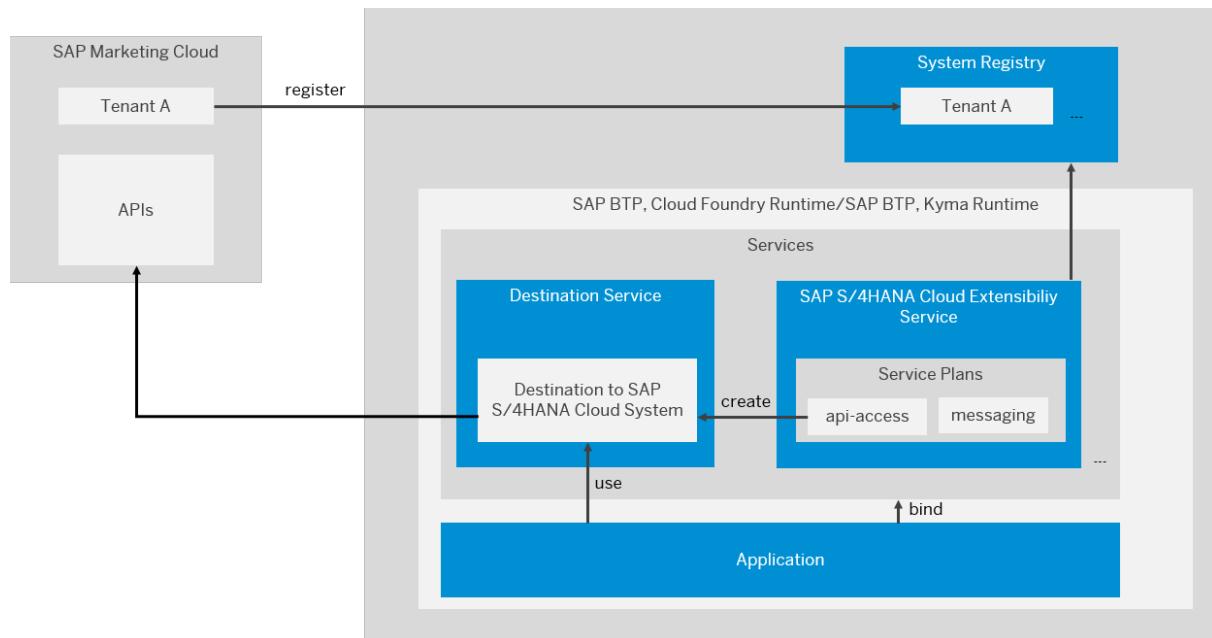
SAP BTP offers a standard way for extending SAP solutions.

You can extend SAP Marketing Cloud in the Cloud Foundry or the Kyma environment without disrupting the performance and the core processes. When building extension applications, you can also benefit from the automation of the integration between the cloud platform and SAP Marketing Cloud.

### Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

The following graphic provides a high-level overview of the integration between SAP BTP and SAP Marketing Cloud:



## Process Flow

To integrate SAP BTP and SAP Marketing Cloud so that you can build extension applications, you have to use the SAP S/4HANA Cloud Extensibility service. All the steps are the same from the SAP BTP side. You register an SAP Marketing Cloud system in the SAP BTP cockpit. Then, you take the generated token and use it to configure the connectivity on the SAP Marketing Cloud side. These are the tasks you need to follow:

## Integrating SAP BTP and SAP Marketing Cloud

Process Step	Related Documentation
<p>1. Connect the SAP Marketing Cloud system you want to extend with the corresponding global account in SAP BTP.</p> <p>During the pairing process you create an registration token which is then used by the SAP Marketing Cloud system tenant administrator to configure the integration on the SAP Marketing Cloud system side.</p>	<p><a href="#">Register an SAP Marketing Cloud System in a Global Account in SAP BTP [page 2006]</a></p>
<p>2. Make the SAP Marketing Cloud system accessible in the subaccounts in SAP BTP in which you want to build your extension applications.</p> <p>To do so, you configure the entitlements and assign the corresponding quota and service plans to the subaccounts where the extension applications will reside for the system you registered in the previous step.</p>	<p><a href="#">Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service [page 2049]</a></p>

### ⓘ Note

You cannot migrate the registered SAP Marketing Cloud systems between global accounts.

If you want to start using another global account, you will have to register your SAP Marketing Cloud systems again.

Process Step	Related Documentation
<p>3. Configure the communication flow.</p> <p>You have the following options:</p> <ul style="list-style-type: none"> <li>Consume the SAP Marketing Cloud APIs (inbound connection) or consume APIs exposed by the extension application from SAP S/4 HANA Cloud (outbound connection)</li> </ul> <p>To do so, you create a service instance of the <a href="#">SAP S/4HANA Cloud Extensibility</a> service using the <a href="#">api-access</a> service plan.</p> <p>During the service instance creation an HTTP destination on a subaccount level is automatically generated in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP Marketing Cloud system. When creating the service instance, you configure the communication arrangement and the authentication type for the connection. The following authentication scenarios for SAP Marketing Cloud are supported:</p> <ul style="list-style-type: none"> <li>Basic Authentication (inbound and outbound connections)</li> <li>OAuth 2.0 SAML Bearer Assertion (inbound connections)</li> <li>OAuth 2.0 Client Credentials (outbound Connections)</li> <li>No Authentication (outbound connections)</li> </ul> <p>Both predefined and custom communication scenarios are supported.</p> <ul style="list-style-type: none"> <li>Enable the consumption of SAP Marketing Cloud events.</li> </ul> <p>If you want to create event-based extensions for SAP Marketing Cloud using the SAP Enterprise Messaging service, you have to create a service instance of the <a href="#">SAP S/4HANA Cloud Extensibility</a> service using the <a href="#">messaging</a> service plan.</p> <ul style="list-style-type: none"> <li>A combination of both</li> </ul>	<ul style="list-style-type: none"> <li><a href="#">Creating a Service Instance to Consume the SAP S/4HANA Cloud APIs [page 2050]</a></li> <li><a href="#">Enable the Consumption of SAP S/4HANA Cloud Events [page 2072]</a></li> </ul>

## 5.4.5 Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment

Use SAP BTP to extend SAP SuccessFactors with extension applications running on the cloud platform.

### Overview

#### ⓘ Note

The SAP SuccessFactors Extensibility service is available for the EU-only access regions. See [Regions](#).

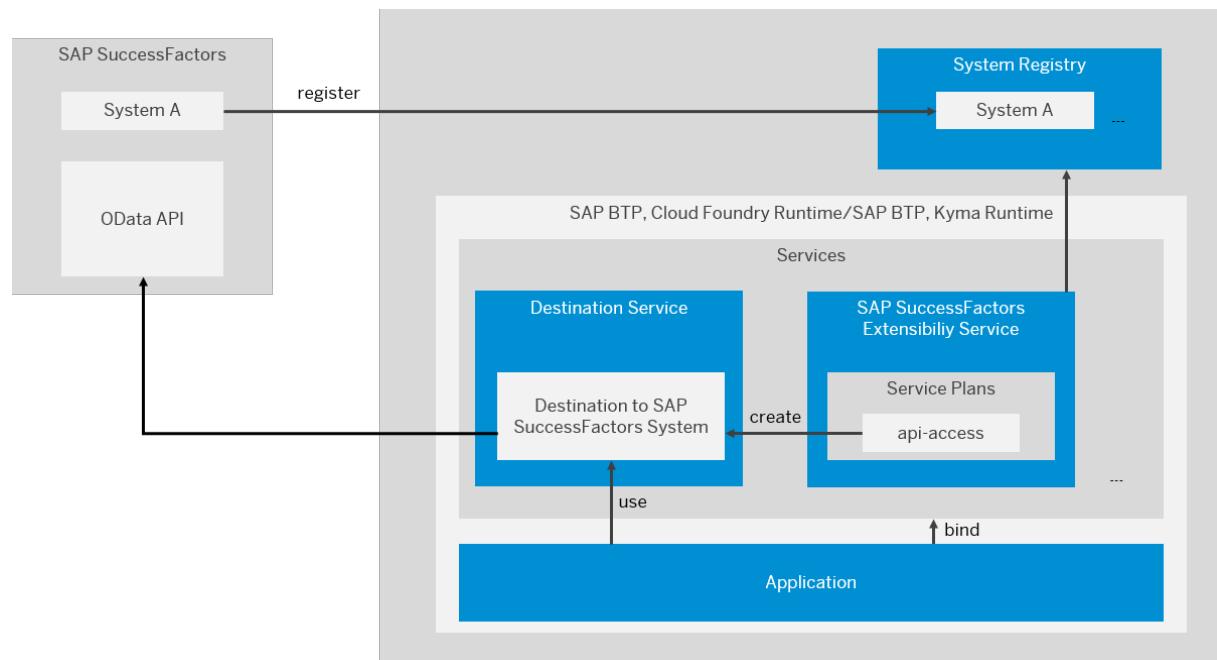
SAP BTP offers a standard way for extending SAP solutions.

You can extend SAP SuccessFactors systems without disrupting the performance and the core processes. When building extension applications, you can also benefit from the automation of the integration between the cloud platform and SAP SuccessFactors.

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

The following graphic provides a high-level overview of the integration between SAP BTP and SAP SuccessFactors:



## Process Flow

To integrate SAP BTP and SAP SuccessFactors so that you can build extension applications, you need to perform the following tasks:

Integrating SAP BTP and SAP SuccessFactors

Process Step	Related Documentation
1. Connect the SAP SuccessFactors system that you want to extend with the corresponding global account in SAP BTP.	<a href="#">Register an SAP SuccessFactors System in a Global Account in SAP BTP [page 2009]</a> .
During the pairing process you create a registration token which is then used by the SAP SuccessFactors system tenant administrator to configure the integration on the SAP SuccessFactors system side.	<p><b>① Note</b></p> <p>You cannot migrate the registered SAP SuccessFactors systems between global accounts.</p> <p>If you want to start using another global account, you will have to register your SAP SuccessFactors systems again.</p>
2. Make the SAP SuccessFactors system accessible in the subaccounts in which you want to build your extension applications.	<a href="#">Configure the Entitlements for the SAP SuccessFactors Extensibility Service [page 2114]</a>
To do so, you configure the entitlements and assign the corresponding quota where the extension applications will reside.	
3. Configure the communication flow.	<a href="#">Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API [page 2115]</a>
To do so, create a service instance of the SAP SuccessFactors Extensibility service using the <i>api-access</i> service plan.	
During the service instance creation an HTTP destination on a subaccount level is automatically generated in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP SuccessFactors system.	
SAP BTP supports the following authentication scenarios for SAP SuccessFactors:	
<ul style="list-style-type: none"><li>• OData access with OAuth 2.0 SAML bearer assertion</li><li>• OData access with OAuth 2.0 SAML bearer assertion with technical user</li></ul>	

Process Step	Related Documentation
<p>4. Configure the Single-Sign On (SSO) between the subaccount in SAP BTP and the SAP SuccessFactors system.</p> <p>To ensure the required security for accessing the applications, you need to configure the single sign-on between the subaccount in SAP BTP and the SAP SuccessFactors system using a SAML identity provider.</p>	<a href="#">Configure Single Sign-On Between a Subaccount in SAP BTP and SAP SuccessFactors [page 2122]</a>
<p>5. If you have performed an automated instance refresh or your cloud operators have performed a manual instance refresh, you need to restore some of the extension configuration settings.</p>	<a href="#">Restore Configuration Settings After an Instance Refresh [page 2127]</a>

### 5.4.5.1 Configure the Entitlements for the SAP SuccessFactors Extensibility Service

Configure the required entitlements to make the SAP SuccessFactors HXM Suite OData APIs of the registered SAP SuccessFactors system accessible in your subaccount in which your extension applications will reside.

#### Prerequisites

- You are an administrator of the global account in SAP BTP.
- The subaccount is in the SAP BTP, Cloud Foundry environment with enabled Cloud Foundry, or Kyma, or both capabilities.
- You have registered an SAP SuccessFactors system. See [Register an SAP SuccessFactors System in a Global Account in SAP BTP \[page 2009\]](#).

#### Context

An entitlement is your right to provision and consume a resource. In other words, the entitlement is the *api-access* service plan that you're entitled to use, and the *sso-configuration* service plan that allows you to register an assertion consumer service in SAP SuccessFactors automatically.

## Procedure

1. In the SAP BTP cockpit, navigate to your global account.
2. In the navigation area, choose *Entitlements* *Entity Assignments*.
3. On the *Entity Assignments* screen, select your subaccount in the *Subaccounts/Directories* field.
4. Choose *Edit*, and then choose *Add Service Plans*.
5. Select the *SAP SuccessFactors Extensibility* service.

### Note

To have the *SAP SuccessFactors Extensibility* service in the list, you need to have registered at least one SAP SuccessFactors system.

6. In the *Service Details* area, select the system you have registered. In the *Available Plans* area, select the *api-access* and *sso-configuration* service plans, and then choose *Add 2 Service Plans*.

## Related Information

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

### 5.4.5.2 Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API

To enable the integration of your extension applications with the SAP SuccessFactors system you have registered in the global account in SAP BTP, you first need to create a service instance of the corresponding service.

## Context

In both Cloud Foundry and Kyma environment, you consume services by creating a service instance. Service instances are created using a specific service plan.

To allow applications running on SAP BTP to consume SAP SuccessFactors HXM Suite OData APIs, you need to create a service instance of the SAP SuccessFactors Extensibility service using the *api-access* service plan.

You create the service instance in your subaccount with the respective environment enabled. When creating the service instance, you configure the authentication type for the communication by specifying the required configurations in a JSON format. SAP BTP supports the following authentication scenarios for SAP SuccessFactors:

- OData access with OAuth 2.0 SAML bearer assertion  
To use this authentication type, you must protect your application. See [Adding Authentication and Authorization \[page 499\]](#).

- OData access with OAuth 2.0 SAML bearer assertion with technical user

The process for creating a service instance depends on the environment you are using:

- For Cloud Foundry environment, see [Create an SAP SuccessFactors Extensibility Service Instance in the Cloud Foundry Environment \[page 2116\]](#).
- For Kyma environment, see [Create an SAP SuccessFactors Extensibility Service Instance in the Kyma Environment \[page 2118\]](#).

### **5.4.5.2.1 Create an SAP SuccessFactors Extensibility Service Instance in the Cloud Foundry Environment**

To enable the integration of your extension applications with the SAP SuccessFactors system you have registered in the global account in SAP BTP, you first need to create a service instance of the corresponding service.

#### **Prerequisites**

- You have registered an SAP SuccessFactors system. See [Register an SAP SuccessFactors System in a Global Account in SAP BTP \[page 2009\]](#).
- Before creating an SAP SuccessFactors Extensibility service instance in the Cloud Foundry environment, see [Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API \[page 2115\]](#).
- Have enabled Cloud Foundry environment for your subaccount. See [Enable Environment or Create Environment Instance \[page 2179\]](#).
- Have configured the entitlements to the SAP SuccessFactors Extensibility service. See [Configure the Entitlements for the SAP SuccessFactors Extensibility Service \[page 2114\]](#).

#### **Context**

In the Cloud Foundry environment, you consume services by creating a service instance. Service instances are created using a specific service plan. The services are offered in the Service Marketplace, from which you can create service instances to provision the reserved resources.

To allow applications running on SAP BTP to consume SAP SuccessFactors HXM Suite OData APIs, you need to create a service instance of the SAP SuccessFactors Extensibility service using the *api-access* service plan.

You create the service instance in a space of your subaccount. When creating the service instance, you configure the authentication type for the communication by specifying the required configurations in a JSON format. SAP BTP supports the following authentication scenarios for SAP SuccessFactors:

- OData access with OAuth 2.0 SAML bearer assertion  
To use this authentication type, you must protect your application. See [Authentication for Applications](#).

- OData access with OAuth 2.0 SAML bearer assertion with technical user

During the creation of the service instance, a destination on a subaccount level with the same name as the service instance name is automatically created in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP SuccessFactors system.

### Note

Make sure that you don't already have a destination with the same name as the service instance. If you do, you will not be able to create the service instance.

## Procedure

1. In the SAP BTP cockpit, navigate to the subaccount in which you want to create a service instance.
2. In the navigation area, choose  [Services](#)  [Service Marketplace](#) .
3. To enable the integration with an SAP SuccessFactors system that you have registered in SAP BTP global account, choose [SAP SuccessFactors Extensibility](#).
4. In the [SAP SuccessFactors Extensibility](#) page, choose [Create](#).
5. In the [New Instance or Subscription](#) wizard:
  - a. In the [Service](#) dropdown list, ensure you have selected the [SAP SuccessFactors Extensibility](#) service.
  - b. In the [Plan](#) dropdown list, select the [api-access](#) service plan.
  - c. In the [Runtime Environment](#) dropdown list, select [Cloud Foundry](#).
  - d. In the [Space](#) dropdown list, select your space. If you haven't create a space yet, you can do it at this point.
  - e. In the [System Name](#) dropdown list, select the SAP SuccessFactors system you have registered.
  - f. In the [Instance Name](#) field, enter a name for your instance. Choose [Next](#).
  - g. (Optional) To define a technical user the access to the SAP SuccessFactors HXM Suite API, specify a JSON file or specify parameters in the JSON format. If you decide to add a technical user later on, you have to delete this service instance and create it again. Choose [Next](#).

For more information about the structure of the JSON file, see [API Access Configuration JSON File \[page 2120\]](#).
- h. Choose [Create](#).

## Results

After you have created the service instance:

- The newly created instance appears in the list of instances in the [Instance and Subscriptions](#) page.
- An HTTP destination on a subaccount level with the same name as the service instance name is automatically generated in this subaccount.

Alternatively, you can use the Cloud Foundry Command Line Interface (cf CLI) to create the service instance using the technical name of the SAP SuccessFactors Extensibility service which is **sap-successfactors-extensibility**.

For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

#### ⓘ Note

You can use the cf CLI to troubleshoot if the creation of the service instance fails. To do that, use this command in the cf CLI:

```
cf service <service_instance_name>
```

## Next Steps

After creating the *SAP SuccessFactors Extensibility* service instance, you have the following options for configuring the connectivity for your extension application:

- Bind the instance to an application, and it will be assigned an access URL and credentials to the corresponding API. For more information about binding applications to service instances, see [Binding Service Instances to Applications](#).
- Consume the automatically generated destination.  
To consume the destination, you use the Destination service. You can either consume the Destination service directly, or configure the application router to consume it.

#### ⓘ Note

The name of the destination is the same as the name of the service instance you have created.

- For more information about consuming the destination service using the application router, see [Application Routes and Destinations](#).
- For more information about consuming the destination service directly, see [Consuming the Destination Service \(Cloud Foundry Environment\)](#).

### 5.4.5.2.2 Create an SAP SuccessFactors Extensibility Service Instance in the Kyma Environment

#### Prerequisites

- Before creating an SAP SuccessFactors Extensibility service instance in the Kyma environment, see [Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API \[page 2115\]](#).
- Have the Kyma environment enabled for the subaccount you are using. See [Create a Kyma Instance \[page 2992\]](#).

- Configure the roles in the Kyma environment. See [Assign Roles in the Kyma Environment \[page 3137\]](#).
- The SAP BTP Operator module is enabled. See [Add and Delete a Kyma Module \[page 3012\]](#)
- Have the entitlements of the SAP SuccessFactors Extensibility service configured. See [Configure the Entitlements for the SAP S/4HANA Cloud Extensibility Service \[page 2049\]](#).

## Context

During the creation of the service instance, a destination on the subaccount level with the same name as the service instance name is automatically created in this subaccount. It contains all instance binding properties which are sufficient to establish connection to the SAP SuccessFactors system.

### ⓘ Note

Make sure that you don't already have a destination with the same name as the service instance. If you do, you will not be able to create the service instance.

## Procedure

1. Navigate to the subaccount for which you want to create an SAP SuccessFactors Extensibility service instance.
2. On the subaccount *Overview* page, in the *Kyma Environment* section, open the Kyma dashboard.
3. Choose *Namespaces* from the left-hand side navigation and open the namespace in which you want to create a service instance.
4. Choose  *Service Management*  *Service Instances*  from the left-hand side navigation.
5. In the *Service Instances* page, choose *Create Service Instance* in the upper right-hand corner. A new dialog opens.
6. Choose the *Advanced* tab and fill in the following fields:
  - Give a meaningful name of the new SAP SuccessFactors Extensibility service instance.
  - In the *Offering Name* field, enter **sap-successfactors-extensibility**, which is the technical name of the SAP SuccessFactors Extensibility service.
  - In the *Plan Name* field, enter **api-access**.
  - Expand *Instance Parameters*.  
To define the authentication type for the API access, specify the parameters listed in this JSON file: [API Access Configuration JSON File \[page 2120\]](#).
7. Choose *Create*.

## Next Steps

After creating the *SAP SuccessFactors Extensibility* service instance, you have to bind the instance to an application, and it will be assigned an access URL and credentials to the corresponding API. See [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#).

### 5.4.5.2.3 API Access Configuration JSON File

Use the authentication type JSON descriptor to define the authentication type for the inbound connectivity to the SAP SuccessFactors HXM Suite OData API.

Parameter	Description
systemName	<p>The name of the system you have registered in a global account in SAP BTP.</p> <p><b> ⓘ Note</b> The system must be in status <i>Registered</i>.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"><li>• Required: Yes</li></ul>

Parameter	Description
technicalUser	<p>The name of the technical user for consuming the SAP SuccessFactors HXM Suite OData API without a logged-in user.</p> <p><b>⚠ Caution</b></p> <p>This property specifies the <i>systemUser</i> property in the destination that is created as a result of creating the SAP SuccessFactors Extensibility service instance.</p> <p>The <i>systemUser</i> is deprecated and will be removed soon. We recommend that you work on behalf of specific (named) users instead of working with a technical user.</p> <p><b> ⓘ Note</b></p> <p>The technical user can be any user with the respective permissions. These permissions depend on the use case and the API you want to access. To find out which permission you need to assign to the technical user, go to <a href="#">SAP Business Accelerator Hub</a>, find the SAP SuccessFactors API you want to access and from the <i>Overview</i> tab go to the <i>Documentation</i> section and open the <a href="#">help.sap.com</a> link. There you find the right information for each API.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> <li>• Allowed characters: [a-zA-Z0-9_-]</li> <li>• Max length: 100</li> </ul>

## Example

```
{
  "systemName": "MY_SYSTEM",
  "technicalUser": "technicalonboarder"
}
```

## 5.4.5.3 Configure Single Sign-On Between a Subaccount in SAP BTP and SAP SuccessFactors

Use this procedure to configure the single sign-on (SSO) between SAP BTP and the SAP SuccessFactors system.

### Prerequisites

You have the *UI and Role Administrator* role assigned to your user. See [Security Administration: Managing Authentication and Authorization \[page 2202\]](#).

### Context

The authentication to SAP SuccessFactors applications is restricted to the authorized users. The identification of a user is verified by the identity provider, as specified by SAML 2.0. The Identity Authentication service stores a list of all users that are allowed to access the SAP SuccessFactors system along with their credentials. The integration between the SAP SuccessFactors and the identity provider is based on a trust configuration. When a user attempts to access SAP SuccessFactors for the first time, the system redirects the user to the identity provider for identification. From then on, the user session is kept active, and the user is no longer prompted for credentials when trying to use the SAP SuccessFactors application. This is called single sign-on (SSO).

To ensure the required security for accessing the applications, you need to configure the single sign-on between the subaccount in SAP BTP and the SAP SuccessFactors system using a SAML identity provider. The single sign-on requires both solutions to be configured as trusted SAML service providers for the identity provider, and at the same time, the identity provider to be configured as trusted identity provider for the two solutions.

If you have an account in the Cloud Foundry environment, you need to set up the single sign-on (SSO) according to this environment.

### Procedure

1. [Configure SAP SuccessFactors as a Trusted Identity Provider in SAP BTP \[page 2123\]](#).
2. [Configure the Subaccount as a Trusted Service Provider in SAP SuccessFactors \[page 2123\]](#).

### **5.4.5.3.1 Configure SAP SuccessFactors as a Trusted Identity Provider in SAP BTP**

Use this procedure to set up the trust configuration of the subaccount in SAP BTP and add SAP SuccessFactors as an identity provider.

#### **Procedure**

1. Download SAML metadata from the SAP SuccessFactors system.
  - a. Go to `https://<sap_successfactors_system>/idp/samlmetadata?company=<company_id>&cert=sha2` where:
    - `<sap_successfactors_system>` is the hostname of your SAP SuccessFactors system
    - `<company_id>` is the ID of your SAP SuccessFactors company
  - b. When you are prompted, save the file on your local file system and change its extension to `.xml`.
2. Register the SAP SuccessFactors identity provider in the SAP BTP cockpit.
  - a. Open the cockpit and navigate to your subaccount.
  - b. Choose  *Security*  *Trust Configuration*.
  - c. Choose *New Trust Configuration*.
  - d. To upload the SAML metadata you downloaded in step 1, choose *Upload*. Browse to the XML file you saved and select it. Some of the fields are automatically filled in.
  - e. In the *Name* field, enter a meaningful name for the trust configuration.
  - f. Save the changes.
3. Make the trust configuration to the SAP SuccessFactors identity provider the only configuration that is active.

### **5.4.5.3.2 Configure the Subaccount as a Trusted Service Provider in SAP SuccessFactors**

To configure the subaccount as a trusted service provider in SAP SuccessFactors, you have to create an SAP SuccessFactors Extensibility service instance using the [sso-configuration](#) service plan.

#### **Prerequisites**

- Register an SAP SuccessFactors System in a Global Account in SAP BTP [page 2009]
- Configure the Entitlements for the SAP SuccessFactors Extensibility Service [page 2114]
- Configure SAP SuccessFactors as a Trusted Identity Provider in SAP BTP [page 2123]

## Context

With the SAP SuccessFactors Extensibility service instance using the [sso-configuration](#) service plan, you can create automatically an assertion consumer service in SAP SuccessFactors for the subaccount in which this service instance is created and a logout URL for each extension application you have deployed in SAP BTP that is extending the functionality of the registered SAP SuccessFactors system. To create an assertion consumer service, you either go to the SAP SuccessFactors Provisioning and create these assertion consumer services manually, or create them automatically in the SAP BTP cockpit via an SAP SuccessFactors Extensibility service instance using the [sso-configuration](#) service plan without loggin in to the Provisioning.

To create an assertion consumer service automatically, you create an SAP SuccessFactors Extensibility service instance of plan [sso-configuration](#) providing all the necessary information in a JSON file. You can have only one service instance of plan [sso-configuration](#) per subaccount.

In the JSON file you provide, in the [systemName](#) parameter, you specify the SAP SuccessFactors system where you want the assertion consumer services to be created. Then, you automatically get an assertion consumer service for the subaccount where the service instance is created and assertion consumer services for each logout URL you provided.

If you want to add or delete assertion consumer services with logout URLs, you have to update the SAP SuccessFactors Extensibility service instance of plan [sso-configuration](#) and provide a JSON file with the respective information. Before updating the service instance, you have the option to view the JSON that was previously passed.

When updating the service instance:

- All the new assertion consumer services with logout URLs will be created in SAP SuccessFactors.
- All the assertion consumer services with logout URLs that were previously created but are not part of the new JSON file will be deleted.
- All the assertion consumer services with logout URLs that were previously created and are part of the new JSON file will be recreated.

When you delete the SAP SuccessFactors Extensibility service instance of plan [sso-configuration](#), all the assertion consumer services with logout URLs created by this service instance will also be deleted.

### ⓘ Note

You cannot bind the service instance of [sso-configuration](#) service plan.

## Procedure

1. In the SAP BTP cockpit, navigate to the subaccount in which you want to create a service instance.
2. In the navigation area, choose [Services](#) [Service Marketplace](#).
3. All services available to you appear.
4. To enable the integration with an SAP SuccessFactors system that you have registered in the global account, choose [SAP SuccessFactors Extensibility](#).
5. In the [SAP SuccessFactors Extensibility](#) page, choose [Create](#).

5. In the [New Instance or Subscription](#) wizard:
  - a. In the *Service* dropdown list, ensure you have selected the *SAP SuccessFactors Extensibility* service.
  - b. In the *Plan* dropdown list, select the *sso-configuration* service plan.
  - c. In the *Runtime Environment* dropdown list, select *Other*.
  - d. In the *System Name* dropdown list, select the SAP SuccessFactors system you have registered.
  - e. In the *Instance Name* field, enter a name for your instance. Choose *Next*.
  - f. To configure the assertion consumer service of the subaccount and the respective applications, specify a JSON file or specify parameters in a JSON format. You can also do this later on. Choose *Next*.

**ⓘ Note**

The content of the JSON file must be up to 5120 characters without spaces.

For more information about the structure of the JSON file, see [Single Sign-On Configuration JSON File \[page 2125\]](#).

- g. Choose *Create*.

## Results

You have the assertion consumer services created in SAP SuccessFactors and have the SSO between your subaccount in SAP BTP and your SAP SuccessFactors system.

You can check the assertion consumer services. To do so, go to the *Admin Center* of your SAP SuccessFactors company and in the *Search* field, enter *Authorized SP Assertion Consumer Service Settings*. This is a read-only information that allows you to verify if all the assertion consumer services that you need are created.

### 5.4.5.3.3 Single Sign-On Configuration JSON File

Use the single sign-on (SSO) configuration JSON file to define the assertion consumer services in SAP SuccessFactors for the subaccount in SAP BTP and for each extension application you have deployed in SAP BTP that is extending the SAP SuccessFactors system functionality.

**ⓘ Note**

The content of the JSON file must be up to 5120 characters without spaces.

Parameter	Description
systemName	<p>The name of the system you have registered in the global account in SAP BTP</p> <p><b>Rules/Guidelines</b></p> <p>The name must be the same as the one defined during the system registration in the global account. See <a href="#">Register an SAP SuccessFactors System in a Global Account in SAP BTP [page 2009]</a>.</p> <div style="background-color: #f0f8ff; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>The system must be in status <i>Registered</i>.</p> </div>
logoutURLs	<p>(Optional) This is a list of the logout URLs. A logout URL is the absolute URL that corresponds to the URL of the application router with the appended value of the <code>logoutEndpoint</code> property. See <a href="#">logout [page 438]</a>.</p> <div style="background-color: #f0f8ff; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>Make sure that the value of the <code>logoutMethod</code> property is set to <b>GET</b> in the <code>xs-app.json</code> file.</p> </div> <div style="background-color: #f0f8ff; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>The <code>SameSite</code> attribute of the Set-Cookie HTTP response header needs to be set to <code>None</code> so that the cookies are sent in all responses to both first-party and cross-origin requests. You also need to set the cookie <code>Secure</code> attribute, because it requires a secure context/HTTPS. See <a href="#">Environment Variables [page 463]</a>.</p> </div> <div style="background-color: #f0f8ff; padding: 10px;"> <p><b>ⓘ Note</b></p> <p>If you don't specify the <code>logoutURLs</code> parameter, only an assertion consumer service for the subaccount in SAP BTP will be created.</p> </div>

### ↳ Sample Code

```
{
  "systemName": "MY_SYSTEM",
  "logoutURLs": [ "https://my-first-site.com/logout", "https://my-second-site.com/logout" ]
}
```

## 5.4.5.4 Restore Configuration Settings After an Instance Refresh

After an instance refresh, you must restore some of your extension integration artifacts and configuration settings for the SAP SuccessFactors target company.

### Prerequisites

- You have a dedicated SAP SuccessFactors company instance.
- You have a registered SAP SuccessFactors system in your global account in SAP BTP.
- You have performed the instance refresh with the *Instance Refresh* tool or your cloud operators have performed an instance refresh manually.

### Context

In SAP SuccessFactors, instance refresh is a procedure in which the data and the settings of a source company instance are copied into another company instance. During the instance refresh the data of the target company instance is deleted and replaced by the data of the source company instance. Therefore, after an instance refresh, if the target company has been integrated with a global account in SAP BTP, the extension integration configuration settings and artifacts in the target company are overwritten by the configuration settings and data coming from the source company.

You can trigger the instance refresh automatically with the *Instance Refresh* tool or your cloud operators can trigger it manually.

If you have performed an instance refresh with the *Instance Refresh* tool, the OAuth clients, and the Assertion Consumer Services (ACS) created by the cloud platform are recreated. In *Provisioning*, the *Extension Management Configuration* page displays the integration details for the target system.

#### ⓘ Note

The OAuth clients are assigned new IDs. This does not affect the configuration settings.

However, if you have the following artifacts, you must reconfigure them in the SAP SuccessFactors target company instance:

- Custom home page tiles
- Permission roles
- Permission groups

#### ⓘ Note

After a manual instance refreshed, no artifacts are reconfigured. If you want to restore the integration settings for the target company instance, you need to recreate the *SAP SuccessFactors Extensibility* service instance.

## Procedure

To restore the configuration settings after an instance refresh, proceed as follows:

- After an automated instance refresh:
  1. If you have configured custom home page tiles in the target company instance, reconfigure them. See [Custom Home Page Tiles](#).
  2. If you have configured permission roles and permission groups, you need to reconfigure them. See [What Are Role-Based Permissions?](#).
- After a manual instance refresh:
  1. For SAP SuccessFactors, Second Half 2021 Release or later, proceed as follows:
    1. Recreate the *SAP SuccessFactors Extensibility* service instance. See [Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API \[page 2115\]](#).
    2. If you have configured permission roles and permission groups, you need to reconfigure them. See [Custom Home Page Tiles](#).
    3. Reconfigure the permission roles and permission groups. See [What Are Role-Based Permissions?](#).

## Results

You have restored the extension integration artifacts and configuration setting of the target company instance.

### 5.4.5.5 Auditing and Logging Information

Here you can find a list of the events that are logged by SAP SuccessFactors Extensibility service. To retrieve the audit logs stored for SAP SuccessFactors Extensibility create a support ticket in component BC-NEO-EXT-SF.

Events written in audit logs

Event grouping	What events are logged	How to identify related log events	Additional information
Creating an integration	Start of create-integration process	<ul style="list-style-type: none"> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "new": "&lt;integration_details&gt;" } ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies an integration of a global account with an SAP SuccessFactors system has been started.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>
	End of create-integration process	<ul style="list-style-type: none"> <li>• "success": true</li> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "new": "&lt;integration_details&gt;" } ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies an integration of a global account with an SAP SuccessFactors system has been completed.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	Failure of create-integration process	<ul style="list-style-type: none"> <li>• "success": true</li> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "new": "&lt;integration_details&gt;" } ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies an integration of a global account with an SAP SuccessFactors system could not be completed.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>
	Start of delete-integration	<ul style="list-style-type: none"> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "old": "&lt;integration_details&gt;" } ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies a deletion of the integration of a global account with an SAP SuccessFactors system has been triggered.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	End of delete-integration	<ul style="list-style-type: none"> <li>• "success": true</li> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "old": "&lt;integration_details&gt;" } ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies a deletion of the integration of a global account with an SAP SuccessFactors system has been completed.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>
	Failure of delete-integration	<ul style="list-style-type: none"> <li>• "success": false</li> <li>• "type": "integration"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "old": "&lt;integration_details&gt;" } ]</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies a deletion of the integration of a global account with an SAP SuccessFactors system could not be completed.</p> <p>The <code>customDetails</code> attribute contains details such as the issuer of the integration token, the global account ID, the name of the SAP SuccessFactors system and the SAP SuccessFactors company ID, the SAP SuccessFactors host name, and other.</p>
Setting up the connectivity	Start of create-inbound-connection process	<ul style="list-style-type: none"> <li>• "type": "inboundConnection"</li> <li>• "attributes": [ { "name": "&lt;object_name&gt;" , "new": "&lt;connection_details&gt;" } ]</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies a creation of a service instance of the SAP SuccessFactors Extensibility service has been triggered.</p> <p>The <code>customDetails</code> attribute contains details such as the subaccount ID, the global account ID, SAP SuccessFactors host name, and other.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	End of create-inbound-connection process	<ul style="list-style-type: none"> <li>• "type": "inboundConnection"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "new": "&lt;connection_detail&gt;" }</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies a creation of a service instance of the SAP SuccessFactors Extensibility service has been completed.</p> <p>The <code>customDetails</code> attribute contains details such as the subaccount ID, the global account ID, SAP SuccessFactors host name, and other.</p>
	Start of delete-inbound-connection	<ul style="list-style-type: none"> <li>• "type": "inboundConnection"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": "&lt;connection_detail&gt;" }</li> <li>• "status": "BEGIN"</li> </ul>	<p>Event that signifies the deletion of a service instance of the SAP SuccessFactors Extensibility service has been started.</p> <p>The <code>customDetails</code> attribute contains details such as the subaccount ID, the global account ID, SAP SuccessFactors host name, and other.</p>
	End of delete-inbound-connection	<ul style="list-style-type: none"> <li>• "success": true</li> <li>• "type": "inboundConnection"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": "&lt;connection_detail&gt;" }</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies the deletion of a service instance of the SAP SuccessFactors Extensibility service has been completed.</p> <p>The <code>customDetails</code> attribute contains details such as the subaccount ID, the global account ID, SAP SuccessFactors host name, and other.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	Failure of delete-inbound-connection	<ul style="list-style-type: none"> <li>• "success": false</li> <li>• "type": "inboundConnection"</li> <li>• "attributes": [ { "name": &lt;object_name&gt;, "old": "&lt;connection_detail_s&gt;" }</li> <li>• "status": "END"</li> </ul>	<p>Event that signifies the deletion of a service instance of the SAP SuccessFactors Extensibility service could not be completed.</p> <p>The customDetails attribute contains details such as the subaccount ID, the global account ID, SAP SuccessFactors host name, and other.</p>

The following information is described in the table columns:

- *Event grouping* - Events that are logged with a similar format or are related to the same entities.
- *What events are logged* - Description of the security or data protection and privacy related event that is logged.
- *How to identify related log events* - Search criteria or key words, that are specific for a log event that is created along with the logged event.
- *Additional information* - Any related information that can be helpful.

## Related Information

[Audit Logging in the Cloud Foundry Environment](#)

### 5.4.5.6 Configuring Backup

You can back up and restore the destination and the trust configuration settings for the integration between the SAP SuccessFactors system and the global account in SAP BTP.

You can back up the following configuration settings:

- The HTTP destination configured during the creation of the SAP SuccessFactors Extensibility service instance. For more, see [Export Destinations](#).
- Your subaccount X.509 certificates. For more information, see [Set up Trust Between Systems](#)
- Your SAP SuccessFactors IdP SAML 2.0 metadata.  
To download your SAP SuccessFactors IdP SAML 2.0 metadata, in the SAP BTP cockpit, navigate to  [Connectivity](#)  [Destinations](#)  [Download IdP Metadata](#) .
- Your identity provider configurations.  
You can back up identity provider configurations with the Identity Provider Management API. For more information see: [Identity Provider Management](#)  API.

## **5.4.5.7 Accessibility Features in SAP SuccessFactors Extensibility Service**

To optimize your experience of SAP SuccessFactors Extensibility service, SAP Business Technology Platform (SAP BTP) provides features and settings that help you use the software efficiently.

### **ⓘ Note**

SAP SuccessFactors Extensibility service runs on the SAP BTP cockpit. For this reason, accessibility features for SAP BTP cockpit also apply. See the accessibility documentation for SAP BTP cockpit on SAP Help Portal at [Accessibility Features in SAP BTP cockpit \[page 2141\]](#).

For more information on screen reader support and keyboard shortcuts, see [Accessibility for End Users](#).

## **5.4.5.8 Troubleshooting for SAP SuccessFactors Extensibility Service**

Information about troubleshooting issues you might experience when extending SAP SuccessFactors on SAP BTP using the SAP SuccessFactors Extensibility service.

### **Cloud Foundry Environment**

These are some issues you might experience when extending SAP SuccessFactors on SAP BTP, Cloud Foundry environment:

- [My SAP SuccessFactors OData Destination Does Not Work \[page 2134\]](#)
- [My SAP SuccessFactors Extensibility Service Instance Failed to Create and It Didn't Say Why \[page 2135\]](#)

### **5.4.5.8.1 My SAP SuccessFactors OData Destination Does Not Work**

SAP SuccessFactors OData destination does not work.

#### **Issue/Symptom**

You are extending SAP SuccessFactors on SAP BTP, Cloud Foundry environment using the SAP SuccessFactors Extensibility service. You are trying to configure the integration between SAP SuccessFactors and SAP Business Technology Platform (SAP BTP) and your SAP SuccessFactors OData destination does not work.

## Solution

To fix your SAP SuccessFactors OData destination, you can try the following:

- Make sure that the names of the destination properties, and their values match the names described in [Create an HTTP Destination in SAP BTP](#).
- Make sure that the values of the *Client Key* and *apiKey* properties match the *API Key* value specified in the SAP SuccessFactors OAuth client.
- Make sure that the X509 certificate specified in the SAP SuccessFactors OAuth client is the same as the one you have in the Cloud Foundry environment. See [Configuring the Extension Application's Connectivity to SAP SuccessFactors](#).
- Make sure that you have the *uaa.user* authorization scope defined in the *xs-security.json* file. See:
  - [Update a Service Instance](#)
  - [UAA Scopes](#)
- Make sure that the *uaa.user* authorization scope is defined in the role template in the *xs-security.json* file. See [Accessing Business Service Data](#).
- Make sure you consume the destination correctly, as described in [Consume the Destination](#).
- Make sure that you use the *user\_token* OAuth grant for retrieving the destination. See [OAuth User Token Exchange Authentication](#).
- Make sure that the name of the destination that you use in your application matches the name of the one created in the SAP BTP cockpit.
- Note that if you have a destination defined as an environment variable, it will overrule the one created in the cockpit.
- Check the logs in Kibana. If you do not have the application logging service, you can add it. See [Access and Analyze Application Logs and Container Metrics](#).
- Make sure that the name of the destination is the same as the name of the SAP SuccessFactors Extensibility service instance you have created. See [Access and Analyze Application Logs, Container Metrics and Custom Metrics](#).

### 5.4.5.8.2 My SAP SuccessFactors Extensibility Service Instance Failed to Create and It Didn't Say Why

My SAP SuccessFactors extensibility service instance failed to create and it didn't say why.

#### Issue/Symptom

You are trying to create an SAP SuccessFactors Extensibility service instance but the creation fails without an error message.

## Solution

This is what you can do:

- You can use the Cloud Foundry command line interface to get the error message by executing the command:

```
cf service <service_instance_name>
```

Check for any error messages there. See [Managing Service Instances](#).

- Can you reach the SAP SuccessFactors company? Check that the tenant is not in maintenance or being updated.
- Check that there is not an already existing destination with the same name as the SAP SuccessFactors Extensibility service instance you are creating. When creating this service instance, a destination with the same name will be automatically created. See [Create an SAP SuccessFactors Extensibility Service Instance in the Cloud Foundry Environment](#).

## 5.4.6 Extending SAP Customer Experience Products in the Kyma Environment

You can configure the integration between SAP BTP and SAP Customer Experience automatically to extend SAP Customer Experience products with applications running on the cloud platform.

### Overview

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

SAP BTP offers a standard way for extending SAP solutions.

You can extend SAP Customer Experience products in SAP BTP, Kyma runtime without disrupting the performance and the core processes. You can build extension applications for SAP Commerce Cloud, SAP Field Management, and SAP Cloud for Customer. When building extension applications, you can also benefit from the automation of the integration between the cloud platform and SAP Customer Experience.

### Process Flow

To integrate SAP BTP and SAP Customer Experience products so that you can build extension applications, you need to perform the following tasks:

Process Step	Related Documentation
1. Add the Application Connector module, and connect the SAP Customer Experience system you want to extend with the corresponding global account in SAP BTP.  During the registration process you create an integration token which is then used in the SAP Customer Experience system for configuring the integration on the SAP Customer Experience side.	<a href="#">Register an SAP Customer Experience System [page 2014]</a>
2. Configure the integration on SAP Customer Experience side.  To do so, you pair the integration token with your SAP Customer Experience system.	<ul style="list-style-type: none"><li>• For an SAP Commerce Cloud system, see <a href="#">steps 2-7 in Retrieving Client Certificate</a>.</li><li>• For an SAP Field Service Management system, see <a href="#">SAP BTP Extensions</a>.</li><li>• For an SAP Cloud for Customer system, see <a href="#">▶ SAP BTP Extensions ➤ Step 2 Configuration in SAP Cloud for Customer Event Notifications</a> in <a href="#">Configure an Event Notification</a> .</li></ul>
3. Assign the SAP Customer Experience system to a formation to enable the API access to the corresponding SAP Customer Experience product's APIs.	<a href="#">Including Systems in a Formation [page 2027]</a>
4. Call a registered external service using Central Application Gateway URL.	<a href="#">Call a registered external service from Kyma</a> ↗

## Related Information

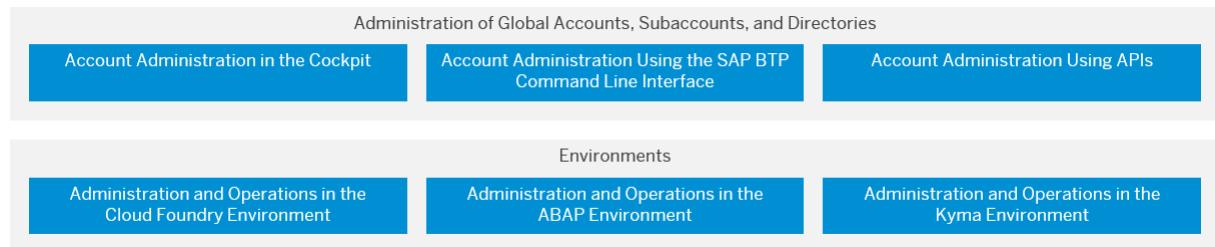
[Getting Started in the Kyma Environment \[page 223\]](#)

[Development in the Kyma Environment \[page 1864\]](#)

[Administration and Operations in the Kyma Environment \[page 2991\]](#)

# 6 Administration and Operations

Administration and operation tasks for SAP BTP include management and configuration of global accounts and subaccounts as well as the operation of applications.



- [Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)
- [Account Administration in the Cockpit \[page 2140\]](#)
- [Administration and Operations in the Cloud Foundry Environment \[page 2422\]](#)
- [Administration and Operations in the ABAP Environment \[page 2536\]](#)
- [Administration and Operations in the Kyma Environment \[page 2991\]](#)
- [Account Administration Using APIs \[page 2378\]](#)

## Administration of Global Accounts, Directories, and Subaccounts

Some of the typical responsibilities of SAP BTP administrators include maintenance of global accounts, management of directories and subaccounts, and configuration of entitlements. To do this, you can use either the SAP BTP cockpit, the SAP BTP command line interface, or APIs.

- [Account Administration in the Cockpit \[page 2140\]](#)
- [Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)
- [Account Administration Using APIs \[page 2378\]](#)

## Administration and Operations in Different Environments

Environments constitute the actual platform-as-a-service offering of SAP BTP that allows for the development and administration of business applications. Each environment offers different approaches to manage administration and operations. For more information, see [Environments \[page 47\]](#).

### Administration and Operations in the Cloud Foundry Environment

In the Cloud Foundry environment, you can manage orgs, spaces, and space quota plans using the SAP BTP cockpit or the SAP BTP command line interface.

Learn more about the different application operations that you can perform in the Cloud Foundry environment.

- [Org Administration Using the Cockpit \[page 2433\]](#)
- [Org Administration Using the Cloud Foundry CLI \[page 2451\]](#)
- [Application Operations in the Cloud Foundry Environment \[page 2490\]](#)
- [Audit Logging in the Cloud Foundry Environment \[page 2501\]](#)

## **Administration and Operations in the ABAP Environment**

When using the ABAP environment, the account administration tasks are performed in the Cloud Foundry environment.

The application operations are specific to the ABAP environment. For more information, see [SAP Fiori Apps in the ABAP Environment \[page 2538\]](#).

## **Administration and Operations in the Kyma Environment**

The administrators of the Kyma environment take care of setting it up and make sure it's ready for developers to work with.

- [Assign Roles in the Kyma Environment \[page 3137\]](#)
- [Create a Kyma Instance \[page 2992\]](#)
- [Kyma Environment Backup \[page 3015\]](#)

## **Related Information**

[Account Model \[page 94\]](#)

## **6.1 Account Administration**

Learn how to manage global accounts, directories, and subaccounts on SAP BTP using different tools.

- [Account Administration in the Cockpit \[page 2140\]](#)
- [Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)
- [Account Administration Using Joule \(Beta\) \[page 2318\]](#)
- [Account Administration Using APIs \[page 2378\]](#)
- [Account Administration Using Infrastructure as Code \[page 2420\]](#)

Once your account model is set up, you can move on to managing the different environments you plan to use:

- [Administration and Operations in the Cloud Foundry Environment \[page 2422\]](#)
- [Administration and Operations in the ABAP Environment \[page 2536\]](#)
- [Administration and Operations in the Kyma Environment \[page 2991\]](#)

## Related Information

[Account Model \[page 94\]](#)  
[Entitlements and Quotas \[page 100\]](#)  
[User and Member Management \[page 104\]](#)  
[Environments \[page 47\]](#)  
[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

### 6.1.1 Account Administration in the Cockpit

Learn about frequent administrative tasks you can perform using the SAP BTP cockpit, such as managing global accounts, directories, subaccounts, entitlements, and members.

#### [Access the Cockpit \[page 2141\]](#)

Learn how to access the SAP BTP cockpit.

#### [Navigate in the Cockpit \[page 2142\]](#)

Learn how to navigate to your global accounts, directories, and subaccounts in the SAP BTP cockpit.

#### [Managing Global Accounts Using the Cockpit \[page 2143\]](#)

Your SAP BTP global account is the entry point for managing the resources, landscape, and entitlements for your departments and projects in a self-service manner.

#### [Managing Directories Using the Cockpit \[page 2165\]](#)

Learn how to organize and manage your subaccounts according to your technical and business needs by using directories in the SAP BTP cockpit.

#### [Managing Subaccounts Using the Cockpit \[page 2173\]](#)

Learn how to structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements by managing subaccounts.

#### [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

When you purchase an enterprise account, you are entitled to use a specific set of resources, such as the amount of memory that can be allocated to your applications.

#### [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#)

Subscribe to multitenant applications from the *Subscriptions* page in the SAP BTP cockpit.

#### [Security Administration: Managing Authentication and Authorization \[page 2202\]](#)

This section describes the tasks of administrators of SAP BTP. Administrators ensure user authentication and assign authorization information to users and user groups.

## Related Information

[Account Model \[page 94\]](#)

## 6.1.1.1 Access the Cockpit

Learn how to access the SAP BTP cockpit.

### Context

Navigate to <https://cockpit.btp.cloud.sap> to access the cockpit. Depending on your geographic location, this URL redirects you to the closest regional cockpit URL.

For more information, see [Cloud Management Tools — Feature Set Overview \[page 129\]](#).

## 6.1.1.1.1 Accessibility Features in SAP BTP cockpit

To optimize your experience of the SAP Business Technology Platform (SAP BTP), SAP BTP cockpit provides features and settings that help you use the software efficiently. Learn more about these features and settings and find out how to use them.

### ⓘ Note

SAP BTP cockpit is based on SAPUI5. For this reason, accessibility features for SAPUI5 also apply. See the accessibility documentation for SAPUI5 on SAP Help Portal at [Accessibility for End Users](#).

For more information on screen reader support and keyboard shortcuts, see [Screen-Reader Support for SAPUI5 Controls](#) and [Keyboard Handling for SAPUI5 Elements](#).

Additionally, SAP BTP cockpit offers a high contrast black theme. See [Use High Contrast Black Theme \[page 2141\]](#).

## Use High Contrast Black Theme

### Context

This feature is recommended for users with need for high contrast themes.

### Steps

1. In the header toolbar choose  [Your Name](#)  [Settings](#).
2. Select the [High Contrast Black](#) theme and click [Apply](#).
3. Once you've saved your changes, the cockpit starts with the theme of your choice.

## 6.1.1.2 Navigate in the Cockpit

Learn how to navigate to your global accounts, directories, and subaccounts in the SAP BTP cockpit.

### Prerequisites

- Sign up for an enterprise or a trial account and receive your logon data. For more information, see [Try Out SAP BTP for Free \[page 144\]](#) or [Sign up for a Customer Account \[page 145\]](#).

### Procedure

1. Navigate to a global account.

- You have only one global account:**

When you log on to the cockpit you're automatically taken into your global account, to the [Account Explorer](#) page.

- You have multiple global accounts:**

When you log on to the cockpit, a dialog opens up containing the list of all global accounts that you're part of. In this dialog, select the global account you want to navigate to.

→ Tip

If you have one global account you usually navigate to, you can choose [Remember my selection](#) in the initial dialog. This means you'll be automatically redirected to your preferred global account after logging on, without seeing the dialog with all the options each time.

If you've chosen a default global account, you can change or remove it anytime. To do so, navigate to the [Account Explorer](#) page of any global account, choose [Switch Global Account](#).

To change it, choose the desired new default global account from the list, select [Save new selection as default global account](#) and choose [Continue](#). Your new default is saved and you're redirected to that global account.

To delete the default global account and go back to seeing the selection dialog after each logon, simply choose the  icon next to your default global account name in the dialog and choose [Close](#).

You can see which global account you are in at any time by looking at the first item in the breadcrumbs. It looks like this:  [`<global account name>`](#)

2. Navigate to a different global account.

- Navigate to the [Account Explorer](#) page at global account level and choose [Switch Global Account](#).
- Use the dropdown menu next to the  [`<global account name>`](#).

3. (Optional) Navigate to directories.

- When you enter your global account, you are by default taken to the [Account Explorer](#) page of that global account. To navigate to a directory, choose the corresponding entry in one of the views in the [Directories & Subaccounts](#) tab.

#### ⓘ Note

If a directory has the **user management** capability enabled, then you can access this directory (or its subdirectories) only if you are a global account admin or you are assigned as a member of the directory. See [Manage Users in Directories \[page 2169\]](#).

The  (*Locked*) icon is shown next to directories that you aren't authorized to access.

4. Navigate to subaccounts.

- a. When you enter your global account, you are by default taken to the *Account Explorer* page of that global account. To navigate to a subaccount, choose the corresponding entry from one of the views.

Once you've entered a subaccount, the breadcrumbs look like this:  <global account name> /  <subaccount name>

#### ⓘ Note

To access a subaccount your user must either have authorizations in the subaccount tenant or, in the case of a subaccount that has the Cloud Foundry environment enabled, have authorizations in the Cloud Foundry org or space of the subaccount. See [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

The  (*Locked*) icon is shown next to subaccounts that you aren't authorized to access.

### 6.1.1.3 Managing Global Accounts Using the Cockpit

Your SAP BTP global account is the entry point for managing the resources, landscape, and entitlements for your departments and projects in a self-service manner.

#### Global Account Settings

As a global account administrator, you can access the global account settings by clicking  (*Global Account Settings*) in the SAP BTP cockpit.

In the *General* tab of the global account settings, you can obtain general information about your global account, including your global account ID, global account name, account type, payment models, contract status, global account subdomain, geographic access restrictions, and when your account was created.

#### ⓘ Note

In this tab, you can change any of the values. The only value that is editable by you is the global account name, and you can do this in the *Account Explorer* page (see [Change the Display Name of Your Global Account \[page 2145\]](#)).

In the *Subaccount Defaults* tab in the global account settings, you can set the supported providers, the default provider, and the default region. These default settings are used when creating a new subaccount in this global account.

## Related Information

[Setting Up Your Account Model](#)

[Managing Directories Using the Cockpit \[page 2165\]](#)

[Managing Subaccounts Using the Cockpit \[page 2173\]](#)

### 6.1.1.3.1 Log On to Your Global Account

Use the SAP BTP cockpit to log on to your global account and start working in SAP BTP.

#### Prerequisites

You have received a welcome e-mail from SAP for your global account. If you don't have a global account administrator or one isn't available, there is a workaround using SAP for Me.

For more information, see [2669325 - How to add a user as a Global Account administrator for SAP Business Technology Platform](#).

#### Procedure

1. Use the link in your Welcome e-mail.

Choose `https://cockpit.btp.cloud.sap` to access the cockpit. Depending on your own geo location this URL will redirect you to the closest regional cockpit URL.

 Note

If single sign-on has not been configured for you, you will have to enter your credentials. You'll find your logon ID in your Welcome e-mail.

2. Choose a global account.

#### Results

The [Account Explorer](#) page for that global account opens.

## Next Steps

- Track and monitor usage of services in your global account:
  - Use the filters to specify which usage information to display in the tables and monthly usage charts. The *Period* filter is applied to the chart display.
  - Some rounding or shortening is applied to large values. Mouse over values in the table to view the exact values in the tooltips.
  - Choose a row in the table to view its historic information in the *Monthly Usage* chart. To display a larger view of a chart, choose the  (*Zoom*) button.
  - To download the information in the *Overview* page to a Microsoft Excel file, choose the *Export* button.
- Access your subaccounts and create new subaccounts.  
See [Create a Subaccount \[page 2173\]](#).

## Related Information

[Account Model \[page 94\]](#)

[Global Accounts \[page 94\]](#)

[Create a Subaccount \[page 2173\]](#)

### 6.1.1.3.2 Change the Display Name of Your Global Account

Change the display name for the global account using the SAP BTP cockpit.

## Prerequisites

You are an admin of the global account that you'd like to edit.

## Context

### → Tip

You can also change the account's display name using the SAP BTP command line interface (btp CLI) with the `btp update accounts/global-account` command. See [Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#).

## Procedure

1. Open your global account in the SAP BTP cockpit.
2. Navigate to the [Account Explorer](#) page.
3. In the *Directories & Subaccounts* view, locate the first entry in your account structure, which is your global account, and click **••• (Actions)**. Then select [Edit](#).
4. In the [Edit Global Account](#) dialog box, enter the new human-readable name for the global account.
5. Save your changes.

## Related Information

[Account Model \[page 94\]](#)

### 6.1.1.3.3 Add Members to Your Global Account

Add users as global account members using the SAP BTP cockpit.

## Prerequisites

- You must be a global account administrator to add other global account members.  
If you don't have a global account administrator or one isn't available, there is a workaround using SAP for Me.  
For more information, see [2669325 - How to add a user as a Global Account administrator for SAP Business Technology Platform](#).
- The users exist in a trusted platform identity provider.  
All users of SAP BTP are stored in identity providers, either in the default or in a custom identity provider. SAP BTP needs a copy of the user, sometimes called a shadow user. You assign the shadow user authorizations to access resources in SAP BTP. When a user authenticates, SAP BTP forwards the request to the identity provider.  
For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

## Context

Assign predefined or custom role collections to users who need to manage or view the global account in SAP BTP cockpit. Examples of predefined role collections include the following:

- [Global Account Administrator](#)
- [Global Account Viewer](#)

For more information about these role collections, see [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

## Procedure

1. Navigate to your global account.
2. Add a user to your global account.

For more information, see [Create Users \[page 2270\]](#).

### ⓘ Note

If the user is already a user in a subaccount and has logged on to SAP BTP cockpit, SAP BTP adds the user at the global account level automatically, but without any role collections.

3. Assign a role collection to the user.

For more information, see [Assign Users to Role Collections \[page 2278\]](#).

## Results

The next time this user logs on to the SAP BTP cockpit, the user can access this global account.

## Related Information

[User and Member Management \[page 104\]](#)

### 6.1.1.3.4 Managing Resource Providers

SAP BTP allows you to connect your global account in the SAP BTP cockpit to your provider account from a non-SAP cloud vendor, and consume remote service resources that you already own and which are supported by SAP through this channel.

### ⓘ Note

The use of this functionality is subject to the availability of the supported non-SAP cloud vendors in your country or region.

## Context

For example, if you are subscribed to Amazon Web Services (AWS) and have already purchased services, such as PostgreSQL, you can register the vendor as a resource provider in SAP BTP and consume this service across your subaccounts together with other services offered by SAP.

SAP BTP currently supports the following vendors and their consumable services:

Cloud Vendor	Supported Services
Amazon Web Services	Amazon Relational Database Service (RDS) - PostgreSQL
Microsoft Azure	Azure Database for PostgreSQL

More information: [PostgreSQL on SAP BTP - Product Documentation](#)

## Configure a New Resource Provider

### Context

To consume the resources provisioned by your provider account, you need to first create a resource provider instance in the cockpit.

### Procedure

1. Navigate to your global account in the cockpit.
2. Choose *Resource Providers* from the left hand-side navigation panel.
3. Choose *New Provider*.
4. Choose the provider from the list of supported vendors.
5. Enter a display name for the provider.

You can create more than one instance of a given resource provider, each with its unique configuration properties. In such cases, the display name (and technical name) should be descriptive enough so that you and developers can easily differentiate between each instance.

6. Enter a unique technical name for the provider.

The technical name can contain only letters (a-z), digits (0-9), and underscore (\_) characters.

This name can be used by developers as a parameter when creating service instances from this provider.

#### Note

After you save the settings for the resource provider, you cannot change its technical name.

7. Choose either *Manually enter provider configuration properties* or *Add provider configuration properties by JSON file*, depending on how you prefer to provide the necessary configuration properties for the given provider.

### → Tip

Each supported vendor has its own unique configuration properties. The [New Resource Provider](#) form provides a description of each property and indicates whether they are mandatory or optional.

If you choose to configure the properties by uploading a JSON file, which is typically provided by the vendor, make sure that the file contains the required configuration properties for connecting to the provider, and is correctly formatted.

### ↔ Sample Code

Sample JSON for configuring Amazon Web Services (AWS) as a resource provider:

```
{  
    "access_key_id": "AWSACCESSKEY",  
    "secret_access_key": "SECRETKEY",  
    "vpc_id": "vpc-test",  
    "region": "eu-central-1"  
}
```

8. Save your changes.

The new resource provider is added to the resource providers table.

## Results

After you configure a new resource provider, its supported services are added as entitlements in your global account. In the  [Entitlements](#) page in the cockpit, you can then allocate the required services and quotas to the relevant directories and subaccounts in your global account.

# Manage Resource Providers and Service Entitlements

## Context

Follow these steps to manage the resource providers that you have already configured in the cockpit.

## Procedure

1. Navigate to your global account in the cockpit.
2. Open the [Resource Providers](#) page.

The page displays the resources providers that you have already configured.

3. In the [Actions](#) columns, you can perform the following for each resource provider:

Action	Description
 <b>Manage Entitlements and Quotas</b>	<p>View the subaccount entitlements and quotas of services that are consumed from a resource provider.</p> <p>This action opens the  <a href="#">Entitlements</a>  <a href="#">Service Assignments</a> page in the cockpit, and is prefILTERed for the selected resource provider.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>→ Tip</b></p> <p>Whenever you need to manage assigned entitlements, you can navigate directly to the <a href="#">Service Assignments</a> or <a href="#">Subaccount Assignments</a> pages in the cockpit without going through the <a href="#">Resource Providers</a> page.</p> </div>
 <b>Edit Resource Provider</b>	<p>Edit the display name, description, and configuration properties of a resource provider.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>ⓘ Note</b></p> <p>You cannot change the technical name of a resource provider.</p> </div>
 <b>Delete</b>	<p>Delete a resource provider.</p> <p>When you delete an existing resource provider, all the services offered by this instance of the resource provider will no longer be available on SAP BTP.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b>ⓘ Note</b></p> <p>You cannot delete a resource provider that already has service entitlements assigned to subaccounts in your global account. To delete the provider, you need to first remove its subaccount assignments in the <a href="#">Subaccount Assignments</a> page.</p> </div>

## Related Information

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

### 6.1.1.3.5 Monitoring Usage and Consumption Costs in Your Global Account

SAP BTP cockpit supports advanced usage and cost monitoring of services in your global account. You can compare the usage and costs of multiple services and subaccounts, see monthly trends, and drill into subaccounts and service plans for detailed information.

- ⓘ Note**

  - The use of the consumption-based commercial model is subject to its availability in your country or region.

- If your global account uses only the subscription-based commercial model, then refer only to information described here about usage data and the [Usage](#) view. You can ignore any information in this topic about costs, cloud credits, and the [Billing](#) view.

The [Costs and Usage](#) page in the cockpit provides a central and flexible experience for the following cost control flows:

- Continuously monitor and analyze the monthly usage and costs of services consumed by subaccounts and directories in your global account.
- Help you to verify your monthly billing statement against actual resource consumption.
- Help you to verify periodic cross-charges between your various business units based on subaccount and directory allocation/distribution.

Check out also the [Enhancing Costs, Usage and Contract Transparency](#) blog for more information.

## Accessing Costs and Usage for Your Global Account

To monitor and track costs and usage in your global account, open the global account in the cockpit and choose [Costs and Usage](#) in the navigation area.

### Note

The [Costs and Usage](#) page is available only if your global account uses the consumption-based commercial model.

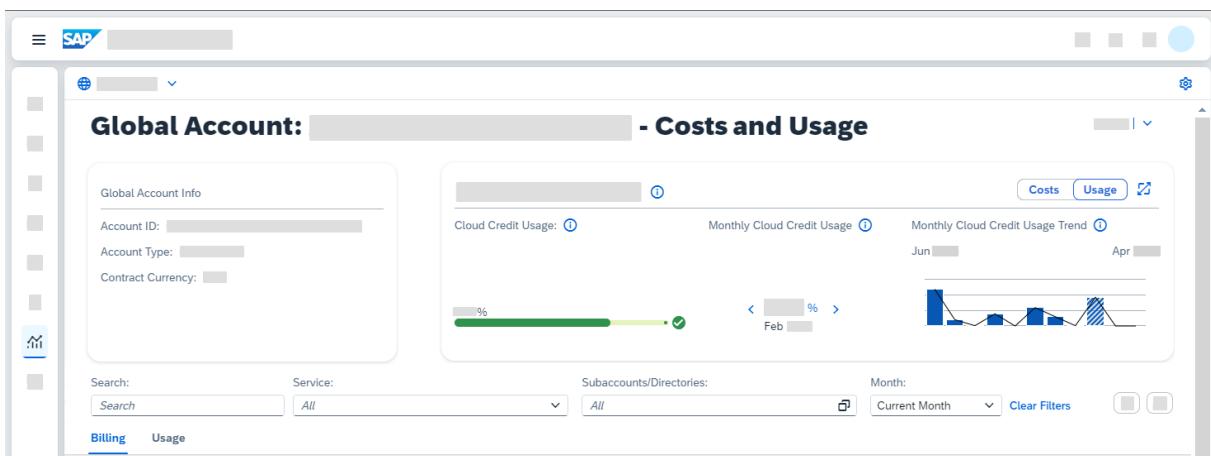
If your global account uses the subscription-based commercial model exclusively, then you must access the [Usage](#) page instead.

The [Usage](#) page is similar to the [Costs and Usage](#) page, except you cannot see any data relating to costs or cloud credits since your eligible services are already prepaid and you aren't charged per usage. See [Using the Usage View \[page 2160\]](#).

## Understanding the Cost and Usage Views

### Global Account and Contract Information

The upper area of the [Costs and Usage](#) page contains two side-by-side cards.



Simplified graphic of the upper area of the Costs and Usage page

Card	What Information is Displayed
<b>Global Account Info</b>	The <a href="#">Global Account Info</a> card on the left displays general info about your global account.
<b>SAP BTP Enterprise Agreement</b>	This is the card on the right; it provides information relating to your cloud-credit usage and costs per month relative to your total cloud credits for the current cloud credits period. It also shows your monthly trend of cloud-credit usage and costs. or
<b>Cloud Platform Enterprise Agreement</b>	This card is displayed only if your global account uses either the SAP BTPEA or CPEA flavor of the consumption-based commercial model.  Use the <a href="#">Usage</a> and <a href="#">Costs</a> buttons in this card to switch the view between usage and cost information.  In this card, you'll see warnings if you're approaching or exceeded your cloud credit limit for the current cloud credits period. Any overages are billed at list price, so make sure to contact SAP if you need more cloud credits.  Note the following: <ul style="list-style-type: none"> <li>Cloud-credit usage and cost information is displayed only for the current cloud credits period. The total contract duration is split into cloud credit periods (typically one year each) and the total cloud credits are divided between these periods.</li> <li>Your cloud credit balance is calculated each month by deducting the corresponding costs of all SAP BTP services for the previous month.</li> <li>If your global account has received a cloud-credit refund at any time during the current cloud credits period, you may see a difference between your total usage/costs and the monthly usage/costs in the chart.</li> </ul>

Pay attention that if your global account uses a combination of the consumption-based commercial model and the subscription-based commercial model, the card on the right shows billing and usage data that is charged solely according to the consumption-based commercial model. In other words, for services that are part of your subscription-based commercial agreement, the billing and usage data in this card applies only to consumption that exceeds your prepaid (subscription) quota. For this excess service usage, you are charged according to your contract for the consumption-based commercial model.

For example, if your subscription contract is entitled to consume a given service at a fixed cost for up to 100 unique site visits, and 151 unique site visits are registered, this card shows data relating only to the 51 visits that have exceeded the prepaid 100 visits.

→ Tip

To see more details about your contract, cloud credit or cost details, click  (*Expand*).

ⓘ Note

To extract your cost and usage data to a spreadsheet document, use the *Export* option. For more information, see [Exporting Usage and Cost Data Information \[page 2161\]](#).

## Filters and Search

Below the two cards, is the filter and search area:

To Do This...	Do This...
Filter the data you want to view	Above the <i>Billing</i> and <i>Usage</i> tabs, apply the filters to choose which services, subaccounts, directories, or billing month you want to view.  You can also use the <i>Search</i> option. When you start typing, the search offers results that apply to service names, plan names, subaccount names, directory names, labels assigned to subaccounts and directories, product IDs (SKU), and metrics. The search offers suggestions only for items that are currently listed.
Share or save the filtered view	<p>→ Tip</p> <p>If you've set up labels in your subaccounts and directories that reflect the distribution of these entities to your company structure or projects, you can easily track and manage cross charges by entering the labels of your subaccount and directory labels in the search.</p> <p>Your filter and search criteria are applied to both <i>Billing</i> and <i>Usage</i> views.</p>
	You can use the  ( <i>Bookmarks</i> ) option to copy the URL of the displayed view and the currently set filters.  You can then save this URL as a bookmark in your browser, or you can share it with a colleague so they can quickly open the same view and filter settings.

## Billing and Usage Views

Below the filter and search area, are the *Billing* and *Usage* views, represented by tabs:

### Billing

Shows the monthly billable service charges in your global account, based on the aggregation of your resource's actual usage.

The aggregation of usage from all your subaccounts is calculated according to the pricing structure and legally billable metric of each service, as mentioned in the [SAP BTP Service Description Guide](#).

#### ⓘ Note

This view relevant only if your global account uses a consumption-based commercial model. If your global account uses only the subscription-based commercial model, then all your eligible services and entitled quota are prepaid.

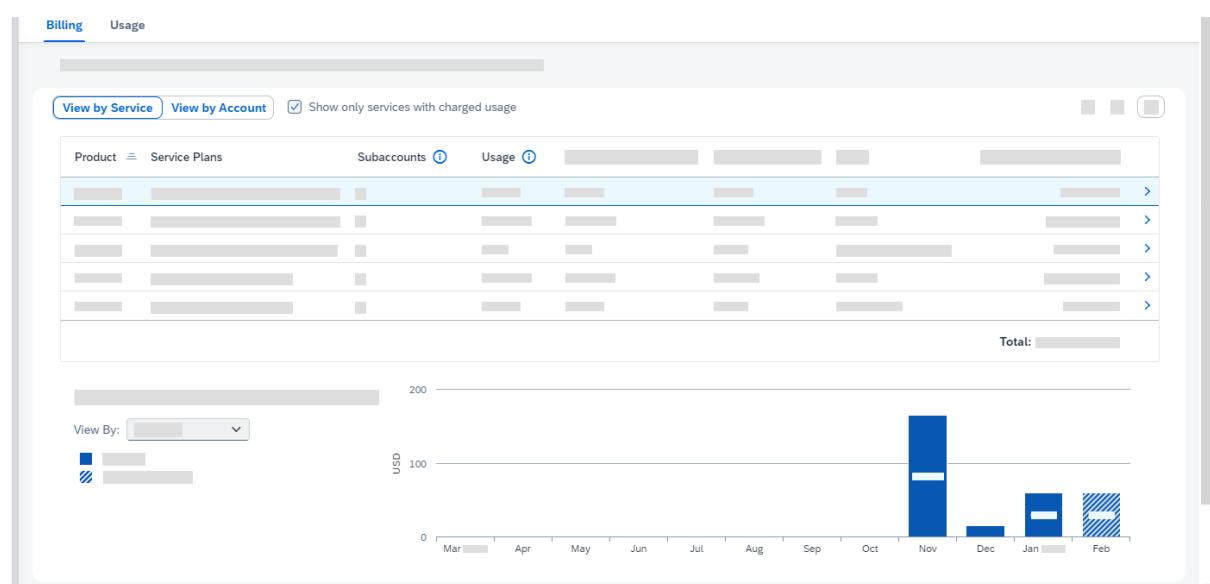
For more information about this view, see [Using the Billing View \[page 2156\]](#).

### Usage

Shows the data representing your actual non-aggregated monthly usage for services consumed in your subaccounts within your global account.

This view applies to both the consumption-based and subscription-based commercial models.

For more information about this view, see [Using the Usage View \[page 2160\]](#).



Simplified graphic of the Billing view of the Costs and Usage page

The *Costs and Usage* page uses the same terminology that is used in your monthly balance statement, which facilitates better contract-to-billing traceability and verification for cost controllers. Get your latest balance statement from [SAP for Me](#).

#### → Remember

- The monthly balance statement, which is provided separately, contains legally binding information regarding your monthly costs. Details about costs on the *Costs and Usage* page in the cockpit are provided for informational purposes only. Any discrepancy between the information displayed in

the cockpit and the information in your balance statement will be resolved in favor of the balance statement.

- Costs are displayed according to your contract currency.
- Global accounts are the only contractual billable entity for SAP BTP. Directories and subaccounts are used as structural entities in global accounts. The usage and cost data displayed for directories and subaccounts are estimations and may differ from the actual global account metrics. Hence, you should use their data only for internal cost estimations.

The relative calculation per billable usage within each subaccount is an estimation only as it is based on certain measures, which in some cases can either be different from the metrics that are presented on the global account level, or that use different formulas than the ones used for billing.

- Cloud credit information and monthly costs apply to all regions used by the subaccounts in your global account. Usage and cost data is updated after your monthly balance statement has been generated.
- Usage data is updated after your monthly balance statement has been generated. For new global accounts, data is updated with the first billing cycle.
- Prices are listed as estimates if a consumable item has not yet been billed.
- In the cost and usage charts, estimated values are used for the period between the last balance statement and the current date. These are displayed as striped bars.  
These estimates are based on resource usage values before computation for billing and might change after the next balance statement is issued. The estimated values are not projected or forecast values.
- Some SAP BTP services report their billing and usage at the global account level, and not at the subaccount level. Such services are listed in the [Costs and Usage](#) page under a reserved technical entity named REPORTED-AT-GLOBAL-ACCOUNT-LEVEL with the ID DEFAULT\_SA.

Here are some useful tips that are common to both the [Billing](#) and [Usage](#) views:

To Do This...	Do This...
Find out what the table columns mean	Click the ⓘ ( <a href="#">More Info</a> ) buttons in the column headings, where available.
Sort the data in ascending or descending order	Click a column heading in the table and choose the sort order you want. Sorting the tables can be useful for determining which service plans and subaccounts have the highest and lowest costs.
Show or hide table columns	Click ⚙️ ( <a href="#">Configure Table Columns</a> ) to configure which table columns you'd like to show and hide. The tables in each view have columns that are hidden by default. Open this setting to check if any of these columns apply to your needs.
Maximize the tables across the entire page	Click ⏕ ( <a href="#">Expand</a> ). When you expand the view, the navigation panel is collapsed and the upper panel containing your global account and contract information is hidden. Click ⏕ ( <a href="#">Collapse</a> ) to go back to the standard view.
Adjust the width of table columns	Drag the column separators to the left or right.

## Using the Billing View

You can use the *Billing* view to display, monitor, and analyze your monthly charges of billable services, which have been consumed by the subaccounts in your global account over the last 12 months.

The billing data represents the aggregation of usage and its rating, which is determined by the pricing structure of each service. You can view the bill items for previous months or the estimated charge for the current month, to monitor your consumption costs.

The parameters for charged usage can include the amount of data processed, the number of users, the duration of use, or the specific functionality or modules used by each service. The specifics of these metrics are outlined in your contract.

→ Tip

This view is available only if your global account uses a consumption-based commercial model.

In the *Billing* view, you can switch between the *View by Service* and *View by Account* perspectives:

#### **View by Service**

This perspective provides an overview of your charged usage across different services and their associated plans, and is useful for verifying billing details.

Here are some useful tips for working in this perspective:

- To drill down and display more details about a particular service, click on an entry in the table or its  (*View Details*) button. When you drill down, a new pane opens and you can see, for example, exactly which subaccounts consume the selected service and the subtotal of charges and usage per directory and subaccount.

In this detailed view, subaccounts are displayed in the context of their account hierarchy. If you want to hide their directories and display the subaccounts as a flat list, choose the *Show subaccounts only* checkbox.

#### → Tip

In this detailed view, you can use the  (*Switch*) button for any subaccount to quickly open the *View by Account* perspective with the specific subaccount selected and displaying the plans it consumes.

You can also click  (*Configure Table Columns*) to display and hide columns.

- To filter the main table for specific services, use the *Service* dropdown list.
- If the display is cluttered by too many service plans or account entities that have no charged usage, you can choose the *Show only services with charged usage* checkbox to hide them.
- To filter the main table for specific subaccounts or directories, use the *Subaccounts/Directories* dropdown list.
- To view a visual chart of the monthly cost trend, cumulative costs, or breakdown of the costliest subaccounts for a specific service plan over the last 12 months, first select the service plan from the main table and then scroll down past the main table to the chart area. Then, select the chart type from the *View By* dropdown list that is located adjacent to the chart area.

#### → Tip

These charts are useful for seeing in which months a service plan has usage and costs, which service plans have increased or decreased usage/costs over time, or the months that have the highest and lowest usage/costs per service plan.

- To view a visual chart with a summary of the costs of all service plans in your global account over the last 12 months, make sure you do not have a specific service plan selected in the main table. Then, scroll past the table to the *Global Account - Cost Summary* chart area. In the *View By* dropdown list, select a chart type; for example, *List Price*, *Cost Breakdown by Service Plans*, or *Cumulative List Price*.

#### → Tip

In the cost breakdown chart, the 8 costliest service plans in your global account over the last 12 months are displayed. Click *Show More* below the chart legend to expand the chart and to view the cost breakdown of additional plans.

See other useful tips, such as table sorting, filtering, and searching, in the [Understanding the Cost and Usage Views \[page 2151\]](#) section.

---

**View by Account** This perspective provides an overview of your charged usage across different subaccounts and directories within your account hierarchy. It is useful for verifying cross charges within your company or organization.

Here are some useful tips for working in this perspective:

- To drill down and display more details about a particular subaccount or directory, click on an entry in the table or its  (*View Details*) button. When you drill down, a new pane opens and you can see, for example, exactly which plans are consumed by the selected subaccount or directory and the subtotal of the charges and usage per plan.

#### → Tip

In this detailed view, you can use the  (*Switch*) button to quickly open the *View by Services* perspective with the specific plan selected and displaying the subaccounts that consume it.

You can also click  (*Configure Table Columns*) to display and hide columns.

- To filter the main table for specific subaccounts or directories, use the *Subaccounts/Directories* dropdown list.
- If your account hierarchy includes directories, you can hide them and display your subaccounts as a flat list by choosing the *Show only subaccounts* checkbox.
- You can quickly expand or collapse the entire account hierarchy by clicking the  (*Expand All*) and  (*Collapse All*) buttons.
- To filter the main table for specific services, use the *Services* dropdown list.
- To view a visual chart of the monthly cost trend, cumulative costs, or breakdown of the costliest service plans for a specific subaccount or directory over the last 12 months, first select the subaccount or directory from the main table and then scroll down past the main table to the chart area. Then, select the chart type from the *View By* dropdown list that is located adjacent to the chart area.

#### → Tip

These charts are useful for seeing in which months a subaccount or directory has usage and costs, which subaccounts or directories have increased or decreased usage/costs over time, or the months that have the highest and lowest usage/costs per subaccount or directory.

When you choose a directory, the cost information applies to all subaccounts under the directory.

- To view a visual chart with a summary of the costs of all subaccounts in your global account over the last 12 months, make sure you do not have a specific subaccount or directory selected in the main table. Then, scroll past the table to the *Global Account - Cost Summary* chart area.  
In the *View By* dropdown list, select a chart type; for example, *List Price*, *Cost Breakdown by Subaccounts*, or *Cumulative List Price*.

#### → Tip

In the cost breakdown chart, the 8 costliest subaccounts in your global account over the last 12 months are displayed. If your global account has more than 8 subaccounts, click

[Show More](#) below the chart legend to expand the chart and to view the cost breakdown of additional subaccounts.

See other useful tips, such as table sorting and searching, in the [Understanding the Cost and Usage Views \[page 2151\]](#) section.

If your global account has subaccounts that have been deleted but have charges in the last 12 months while they were still active, they'll still be listed. Such subaccounts are marked with [\(deleted\)](#) after their name. If these subaccounts were originally located under a directory, you'll see them listed directly under the root global account.

If your global account uses both a subscription-based and a commercial-based model, the [Usage](#) column shows the combined total usage of services, both subscription-based and consumption-based. Note that:

- The subscription-based consumption includes usage of services that falls within the prepaid quota that is specified in your subscription-based commercial model agreement. This part of the combined usage is shown in the [Prepaid Quota](#) column. There will be no additional billing for consumption of the services that falls within this prepaid quota.
- The consumption-based usage includes the usage of services that exceeds the prepaid subscription-based quota. This part of the combined usage is shown in the [Charged Usage](#) column. For this usage, you'll be charged based on the terms outlined in your consumption-based commercial model agreement.

## Using the Usage View

Use the [Usage](#) view to display, monitor, and analyze the distribution of the actual (also called raw) monthly usage metrics for services consumed by subaccounts in your global account over the last 12 months.

### ⓘ Note

The [Usage](#) view displays data prior to aggregation for billing and can contain metrics that will not be aggregated since they are always free or belong to free services. Hence, you may notice some discrepancies when comparing the data in the [Usage](#) view to the overall usage data shown in the [Billing](#) view. Billable usage data is aggregated at the global account level and then processed according to accounting formulas to generate your monthly billing statement.

Pay attention also to the following:

- If your global account uses only a consumption-based commercial model, such as SAP BTP Enterprise Agreement (SAP BTPEA), Cloud Platform Enterprise Agreement (CPEA), and Pay-As-You-Go for SAP BTP, this view includes all usage data, including non-rated services, such as free service plans.
- If your global account uses both the consumption-based and subscription-based commercial models, this view combines all usage data, which falls under your consumption-based plans, prepaid subscription quota, and non-rated data sets, such as free service plans. You can see the distribution of charged and prepaid usage and costs in the [Billing](#) view.

To Do This...	Do This...
Drill down and display more details about a particular service	<p>Click on an entry in the table or its <a href="#">(View Details)</a> button. When you drill down, a new pane opens and you can see, for example, which subaccounts are consuming the selected service and the subtotal of usage per directory and subaccount.</p> <p>In this detailed view, subaccounts are displayed in the context of their account hierarchy. If you want to hide their directories and display the subaccounts as a flat list, choose the <a href="#">Show subaccounts only</a> checkbox.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>→ Tip</b></p> <p>In this detailed view, you can click  <a href="#">(Configure Table Columns)</a> to display and hide columns.</p> </div>
Filter by specific services	<p>Choose one or more in the <a href="#">Service</a> dropdown list.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>→ Tip</b></p> <p>If the display is cluttered by too many service plans that have no usage, choose the <a href="#">Show only used services</a> checkbox to quickly hide them.</p> </div>
Filter by specific subaccounts and directories	<p>Choose one or more subaccounts or directories in the <a href="#">Subaccounts/Directories</a> dropdown list.</p>
View the monthly usage trend of a specific service plan over the last 12 months	<p>Select the service plan from the main table and scroll past the table to the chart area. Then, choose <a href="#">Actual Usage</a> in the <a href="#">View By</a> dropdown list that is located adjacent the chart area.</p> <p>This chart is useful for seeing changes in usage trends over time, including the months with the highest and lowest usage per service.</p>
View a monthly breakdown of subaccounts that have used a specific service plan the most over the last 12 months	<p>Select the service plan from the main table and scroll past the table to the chart area. Then, choose <a href="#">Usage Breakdown by Subaccounts</a> in the <a href="#">View By</a> dropdown list.</p> <p>The top 8 subaccounts that use the selected plan the most are displayed initially. If there are more than 8 subaccounts that use the plan, click <a href="#">Show More</a> to expand the chart and to view the breakdown of additional subaccounts.</p>
<p><b>→ Tip</b></p> <p>See other useful tips, such as table sorting, filtering, and searching, in the <a href="#">Understanding the Cost and Usage Views [page 2151]</a> section.</p>	

## Exporting Usage and Cost Data Information

To export your usage and cost data to a Microsoft Excel spreadsheet document, use the [Export](#) button menu in the upper area of the page.

## ⓘ Note

Cost information is exported only if your global account uses the consumption-based commercial information.

The *Export* button menu has various options depending on the period and type of data that you want to export:

- If you want to export all data for the last 12 months, click the main *Export* button.
- If you want to export all data for the last 12, 24, or 36 months, click the button's dropdown menu and choose the relevant period.
- If you want to export only the data that you've currently filtered in the main table for the current month or last 12, 24, or 36 months, choose ► *Export* ► *Custom* ▾.

## → Tip

Use this option also if you want to export all your cost and usage data for the current month without considering any currently set filters. To do this, deselect the *Export only filtered data* checkbox in the *Custom Export* dialog box.

In the *Custom Export* dialog box, you can also choose any label names (keys) that you want to add as dedicated columns to the Subaccount Costs by Service and Actual Usage sheets (tabs) in the exported Microsoft Excel file. Start typing to list the available label names for your existing subaccounts and directories. In the exported spreadsheet, the values for each selected label name will be displayed in the in the data rows under their respective label column.

The exported document contains several sheets (tabs). The sheets that are included depend on the commercial model that is used by your global account as shown in the following table:

Sheet Name	Description	Commercial Model	
		Subscription-Based	Consumption-Based
<b>Global Account Info</b>	Provides general information about your global account. If there is a cloud-credit balance for the global account, then its cloud-credit usage, per month as a percentage of your total cloud credits for the current cloud credits period, is also shown.	Yes	Yes
<b>Global Account Costs</b>	Allows you to view total monthly usage data and costs for all billable services and plans consumed at the level of your global account.  The items listed are all the billed items that are created in the accounting system and deducted from the cloud-credit balance of your global account.	No	Yes

Sheet Name	Description	Commercial Model	
		Subscription-Based	Consumption-Based
<b>Subaccount Costs by Service</b>	<p>Allows you to view monthly usage data and costs of all billable services consumed by plan and subaccount.</p> <p>All subaccount calculations are estimated and proportionate to the total global account usage:</p> $[\text{Subaccount usage} / \text{Global account usage} \times \text{Rate plan per SKU}]$ <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>Global accounts are the only contractual billable entity for SAP BTP. Directories and subaccounts are used as structural entities in the global account, and their usage and cost data should only be used for your internal cost estimations. The relative calculation per billable usage within each subaccount is an estimation only as it is based on certain measures which in some cases can either be different from the metrics that are presented in the <a href="#">Global Account Costs</a> tab, or which use different formulas than the ones used for billing.</p> </div>	No	Yes
<b>Actual Usage</b>	<p>Allows to you to view the actual monthly usage data of all consumed services by plan, subaccount, and space.</p> <p>Actual or raw usage data is different to the billed usage that is used in your billing document, and includes non-billable services.</p> <p>The aggregated usage for each service is based on a formula that is specific to each service, for example, MIN, MAX, or AVG.</p>	Yes	Yes
<b>Labels</b>	<p>Lists the user-defined labels that are assigned directly to each subaccount that has usage data. Also lists the labels that are inherited by the subaccounts from their parent directories, and the IDs of those subaccounts.</p> <p>This sheet does not display any usage data; however, you can use this information to determine usage by label by extrapolating the subaccount usage data in the other sheets to the labels assigned to the subaccounts in this sheet.</p>	Yes	Yes

## Related Information

[Commercial Models \[page 86\]](#)

[Trial Accounts and Free Tier \[page 82\]](#)

[Using Free Service Plans \[page 91\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

[View Subaccount Usage Analytics \[page 2182\]](#)

[View Directory Usage Analytics \[page 2171\]](#)

[Labels \[page 98\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Monitoring Usage Information Using APIs of the SAP Usage Data Management Service \[page 2409\]](#)

### 6.1.1.3.6 Configure Legal Information

Administrators can define legal links per enterprise global account in the SAP BTP cockpit.

#### Prerequisites

You have the Administrator role for the global account for which you'd like to define legal links.

#### Context

You can define the privacy link relevant for a global account so the members of this global account can view this information.

#### Procedure

1. Choose the global account for which you'd like to make changes.
2. In the left-hand navigation, choose [Legal Information](#) [Configure Privacy Link](#) .
3. Enter the relevant URL.
4. Save your changes.

The links you configured are available in the [Legal Information](#) menu.

## Related Information

[Navigate in the Cockpit \[page 2142\]](#)

[Global Accounts](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

### 6.1.1.4 Managing Directories Using the Cockpit

Learn how to organize and manage your subaccounts according to your technical and business needs by using directories in the SAP BTP cockpit.

#### Context

#### Procedure

1. Learn how to [Create a Directory \[page 2166\]](#).
2. Learn how to [Manage the Account Explorer Hierarchy \[page 2168\]](#).
3. Learn how to [Manage Users in Directories \[page 2169\]](#).
4. Learn how to [View Directory Usage Analytics \[page 2171\]](#).

## Related Information

[Directories \[page 96\]](#)

[Managing Subaccounts Using the Cockpit \[page 2173\]](#)

## 6.1.1.4.1 Create a Directory

Create a directory using the SAP BTP cockpit to organize and manage your subaccounts. For example, you can group subaccounts by project, team, or department.

### Context

#### → Recommendation

Before creating your subaccounts, we recommend you learn more about [Setting Up Your Account Model](#).

For more information about directories, see [Directories \[page 96\]](#).

### Procedure

1. In your global account, navigate to the [Account Explorer](#) page.
2. In the [Create](#) dropdown, choose [Directory](#).

#### → Tip

You can also create a directory by selecting [Create Directory](#) from the [Actions](#) menu button of the global account or an existing directory.

3. In the [Create Directory](#) dialog box, enter a display name for your new directory and choose a parent. The parent can be the global account or another directory.
4. **Optional:** Under [Advanced](#), choose the [Enable entitlement management](#) option if you want to manage the entitlements that are used by the subaccounts under this directory. This allows you to reserve quota for service plans so that they cannot be assigned to other subaccounts or directories. For more information, see [Entitlements and Quotas \[page 100\]](#) and [Configure Entitlements and Quotas for Directories \[page 2188\]](#).

You can enable this feature later from the [Entitlements > Entity Entitlements](#) page or by editing the directory in the [Account Explorer](#) page.

If you don't enable this feature, then subaccounts in this directory use the entitlements and quota from the global account level.

#### ⓘ Note

Only a single directory in any given directory path can have the user management (see next option) and/or entitlement management features enabled.

For example, if you have 3 stacked directories in your account hierarchy and the middle directory has both the user and entitlement management features enabled, then neither of these features can be enabled for its parent or child directory since these two directories are in the same direct path as the middle directory.

5. **Optional:** Under *Advanced*, choose the *Enable user management* option if you want to assign specific users as administrators or viewers of this directory. For more information, see [Manage Users in Directories \[page 2169\]](#).

You can enable this feature later from the directory's *Users* page or by editing the directory in the *Account Explorer* page.

If you don't enable this feature, then any user that has access to the directory in the account hierarchy can view and work in it.

#### ⓘ Note

- The user management feature can be enabled only in combination with the entitlement management feature on the same directory in given path.
- Only a single directory in any given directory path can have the user management and/or entitlement management features enabled.  
For example, if you have 3 stacked directories in your account hierarchy and the middle directory has both the user and entitlement management features enabled, then neither of these features can be enabled for its parent or child directory since these two directories are in the same direct path as the middle directory.

6. **Optional:** Under *Advanced* *Labels*, assign labels to the directory to make organizing and filtering your directories and their subaccounts easier. For more information, see [Labels \[page 98\]](#).

#### → Tip

- When adding multiple values to a label, press `Enter` after each value.
- When you start typing, any matching label names and values that are already assigned to other directories in your global account are offered as suggestions.

#### ⓘ Note

All subaccounts in the path of this directory also inherit all labels that are assigned to this directory.

7. Review the details of your directory and choose *Create* to finalize the process.

## Results

Your directory is created.

## Next Steps

Once you've created a directory, you can perform the following actions:

- Edit the directory - you can edit the name, description, entitlements, and labels of the directory.
- Add subaccounts to the directory - [Manage the Account Explorer Hierarchy \[page 2168\]](#).
- Assign users and entitlements to the directory (if you've enabled the user and entitlement management features, respectively).

- Delete the directory - you can only delete a directory that does not contain any subaccounts or subdirectories.

## Related Information

[Directories \[page 96\]](#)

[Manage the Account Explorer Hierarchy \[page 2168\]](#)

[Manage Users in Directories \[page 2169\]](#)

[Configure Entitlements and Quotas for Directories \[page 2188\]](#)

[View Directory Usage Analytics \[page 2171\]](#)

[Labels \[page 98\]](#)

### 6.1.1.4.2 Manage the Account Explorer Hierarchy

Create an account structure by creating a hierarchy of directories and subaccounts using the SAP BTP cockpit. Add, move, and delete subaccounts and directories in your structure.

#### Context

Here, you'll learn how to:

- Add directories and subaccounts to a directory
- Move directories and subaccounts in the account structure
- Delete directories and subaccounts from the account structure
- Export the account structure to Microsoft Excel format.

A subaccount moves with its assigned service plans and quota. The entitlements of the source and target directories are adjusted accordingly.

If any entitlement for a plan in the target directory is configured with the auto-assign option, it is then automatically applied to the moved subaccount. Automatic quota assignments are subject to available quota in the target directory.

You can create a hierarchical structure that is up to 7 levels deep. The highest level of a given path is always the global account and the lowest is a subaccount, which means that you can have up to 5 levels of directories between the global account and the lowest level subaccount.

#### Procedure

- Add directories and subaccounts to a directory.

- In the *Create* dropdown, choose *Directory*.  
When creating the new directory, you choose its location in the account structure by selecting its parent directory or the global account directly in the creation dialog box.
- You can also create a directory by selecting *Create Directory* from the *Actions* menu button for the global account or a specific parent directory.
- Move subaccounts in the account structure.
  - Click *Actions* and select *Edit*. Then select a new parent.
  - Click *Actions* and select *Move*. Then select a new parent.
  - Click *Move* to move subaccounts by drag and dropping.
- Delete directories and subaccounts from the account structure.
  - Click *Actions* and select *Delete*.
  - To delete a subaccount, you can also navigate into the subaccount and click *Delete Subaccount*.
- Export the account structure to Microsoft Excel format.
  - Click *Export*.  
The exported Microsoft Excel file contains a sheet (tab) called *Account Structure*, which lists all the subaccounts and directories in the account structure, and includes information about each entity. Each subaccount and directory displays its name, ID, description, path, creation/change dates, and the user that created it.  
Subaccounts also display information about their environment, provider, region, used for production setting, and whether the beta feature option is enabled. Directories also display whether they are configured to manage users and entitlements.

## Related Information

[Directories \[page 96\]](#)

[Create a Directory \[page 2166\]](#)

[Create a Subaccount \[page 2173\]](#)

[Delete a Subaccount \[page 2184\]](#)

### 6.1.1.4.3 Manage Users in Directories

Manage members in your directory using the SAP BTP cockpit.

#### Prerequisites

- You're a directory administrator.
- User management is enabled for this directory.

If the feature isn't enabled already, you can do so using one of the following methods:

- Go to the *Account Explorer* page, edit the directory, and then under *Advanced*, choose the *Enable User Management* option.

- Navigate into the directory from the [Account Explorer](#) page, and then in the [Users](#) page, choose the [Enable Entitlement and User Management](#) option (if entitlement management is already enabled for the directory, then choose the [Enable User Management](#) option instead).

### Note

- The user management feature can be enabled only in combination with the entitlement management feature on the same directory in given path.
- Only a single directory in any given directory path can have the user management and/or entitlement management features enabled. See [Configure Entitlements and Quotas for Directories \[page 2188\]](#).

For example, if you have 3 stacked directories in your account hierarchy and the middle directory has both the user and entitlement management features enabled, then neither of these features can be enabled for its parent or child directory since these two directories are in the same direct path as the middle directory.

- Your platform user exists in a trusted identity provider.

All users of SAP BTP are stored in identity providers, either in the default or in a custom identity provider. SAP BTP needs a copy of the user, sometimes called a shadow user. You assign the shadow user authorizations to access resources in SAP BTP. When a user authenticates, SAP BTP forwards the request to the identity provider.

For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

## Context

Assign predefined or custom role collections to users who need to manage or view the directory in SAP BTP cockpit. Examples of predefined role collections include the following:

- [Directory Administrator](#)
- [Directory Viewer](#)

For more information about these role collections, see [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

## Procedure

1. Navigate to your directory.
2. Add a user to your directory.

For more information, see [Create Users \[page 2270\]](#).

3. Assign a role collection to the user.

For more information, see [Assign Users to Role Collections \[page 2278\]](#).

## Results

The next time this user logs on to the SAP BTP cockpit, the user can access this directory.

## Related Information

[Working with Users \[page 2265\]](#)

[Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)

### 6.1.1.4.4 View Directory Usage Analytics

You can explore, compare, and analyze all your actual usage data for the services and applications that are available in your directory.

To monitor and track usage in a directory, open your directory from the [Account Explorer](#) in the SAP BTP cockpit, and then go to the [Usage Analytics](#) page in the navigation area.

#### Note

This cockpit page is available only for directories that have user management enabled.

You must also be assigned as an admin of the directory. For the more information, see [Create a Directory \[page 2166\]](#) and [Manage Users in Directories \[page 2169\]](#).

## Views in the Directory 'Usage Analytics' Page

The directory [Usage Analytics](#) page contains two views that display the same usage information; however, your entry point is different – either from a service/subscription or a directory/subaccount:

View	Description
<b>Service Usage</b>	Displays high-level usage information for a selected service or business application subscriptions according to the directory from which you accessed the <a href="#">Usage Analytics</a> page. You can also drill down and view usage data for specific subdirectories and subaccounts that are located in this directory.
<b>Directory/Subaccount Usage</b>	Displays high-level usage information for all services and business application subscriptions according to the directory from which you accessed the <a href="#">Usage Analytics</a> page. You can also drill down and view usage data for specific subdirectories and subaccounts that are located in this directory.

Information is shown only for those services and subscriptions whose metered consumption in the directory is greater than zero.

Usage values are updated every 24 hours. Usage data displayed in this cockpit page refers to actual usage, not billed usage.

The [Space](#) filter is available only when you've selected a subaccount and the subaccount has the Cloud Foundry environment enabled.

→ Tip

- If your directory is in a global account that uses the consumption-based commercial model, you can view information about your billed usage for billable services in your global account's [Usage Analytics](#) page. The global account's [Usage Analytics](#) page also provides information about cloud-credit usage. For more information, see [Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#).
- You can also view usage information by subaccount directly in a subaccount's [Usage Analytics](#) page. For more information, see [View Subaccount Usage Analytics \[page 2182\]](#).

## Working with the Tables and Charts

Both the [Service Usage](#) and [Directory/Subaccount Usage](#) views display the usage information in tabular and chart formats.

- The tables present accumulated usage values based on the aggregation logic of each service plan and its metric over the selected time period. The usage values are broken down by resource.
- The charts present usage values for one or more service plans that you select in the adjacent tables.

Select a table row to display the usage information in the chart. You can also compare usage between service plans in the same service by selecting multiple rows. To compare usage, the rows items must also share the same metric.

In the charts, you can view the data as a column chart or as a line chart.

To display a larger view of a chart, choose the  [\(Zoom\)](#) button.

Some rounding or shortening is applied to large values. You can mouse over values in the table to view the exact values as tooltips.

## Related Information

[Org Administration Using the Cockpit \[page 2433\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#)

[What Is the Consumption-Based Commercial Model? \[page 87\]](#)

[View Subaccount Usage Analytics \[page 2182\]](#)

[Monitoring Usage Information Using APIs of the SAP Usage Data Management Service \[page 2409\]](#)

[Manage Users in Directories \[page 2169\]](#)

## 6.1.1.5 Managing Subaccounts Using the Cockpit

Learn how to structure a global account according to your organization's and project's requirements with regard to members, authorizations, and entitlements by managing subaccounts.

### Context

To learn more about the concept of subaccounts, see [Subaccounts \[page 95\]](#).

### Procedure

1. Learn how to create subaccounts.  
[Create a Subaccount \[page 2173\]](#)
2. Learn how to manage users.  
[Add Members to Your Subaccount \[page 2177\]](#)
3. Learn how to administer your subaccount.
  - [Change Subaccount Details \[page 2180\]](#)
  - [View Subaccount Usage Analytics \[page 2182\]](#)
  - [Delete a Subaccount \[page 2184\]](#)

### 6.1.1.5.1 Create a Subaccount

Create subaccounts in your global account using the SAP BTP cockpit.

### Create a Subaccount

#### Prerequisites

You are a global account administrator.

##### Note

If you are creating a subaccount in an entitlement-managed or user-managed directory, then you must be either a global account administrator or a directory administrator. See [Configure Entitlements and Quotas for Directories \[page 2188\]](#) and [Manage Users in Directories \[page 2169\]](#).

## → Recommendation

Before creating your subaccounts, we recommend you learn more about [Setting Up Your Account Model](#).

## Context

You create subaccounts in your global account. Once you create a new subaccount, you see a tile for it in the global account view, and you are automatically assigned to it as an administrator.

## Procedure

1. From your global account, navigate to the [Account Explorer](#) page.
2. Select  [Create](#)  to create a subaccount from scratch.

### → Tip

You can also create a subaccount from scratch by selecting [Create Subaccount](#) from the  ([Actions](#)) context menu of the global account or a directory.

### → Tip

Alternatively, you can create a new subaccount from an existing subaccount. For more information, see the [Create a Subaccount from an Existing Subaccount](#) procedure below.

3. Specify a display name.
4. **Optional:** Enter a description.
5. Enter a subdomain for your subaccount. This will become part of the URL for accessing applications that you subscribe to from this subaccount.

### ⓘ Note

The subdomain can contain up to 63 characters such as letters, digits and hyphens (not allowed in the beginning or at the end), and must be unique across all subaccounts in the same region. Uppercase and lowercase letters can be used, however that alone does not qualify as a means to differentiate subdomains (e.g. **SUBDOMAIN** and **subdomain** are considered to be the same).

6. Choose the desired region for your subaccount.
7. Select a parent for your subaccount. The parent can either be the global account or a directory.
8. If your subaccount is to be used for production purposes, select the [Used for production](#) option.

### ⓘ Note

This does not change the configuration of your subaccount. Use this flag for your internal use to operate your production subaccounts in your global account and systems more efficiently. Your cloud operator may also use this flag to take appropriate action when handling incidents related to mission-critical accounts in production systems.

You can change your selection at any time by editing the subaccount properties. Do not select this option if your account is used for non-production purposes, such as development, testing, and demos.

9. **Optional:** To use beta services and applications in the current subaccount, under *Advanced*, select [Enable beta features](#).

 **Caution**

You shouldn't use SAP BTP beta features in subaccounts that belong to productive enterprise accounts. For more information, see [Important Disclaimers and Legal Information](#).

 **Note**

Once you have enabled this setting in a subaccount you cannot disable it.

10. **Optional:** Under  , assign labels to the subaccount to make organizing and filtering your subaccounts easier. For more information, see [Labels \[page 98\]](#).

 **Tip**

- When adding multiple values to a label, press `Enter` after each value.
- When you start typing, any matching label names and values that are already assigned to other subaccounts in your global account are offered as suggestions.

11. Save your changes.

## Next Steps

Once your new subaccount is set up, navigate to it to enable the environment that you wish to use and add users.

# Create a Subaccount from an Existing Subaccount

## Prerequisites

You are a global account administrator.

## Context

Instead of creating a subaccount from scratch, you can create one from an existing subaccount that is similar to the one you want to create. This saves you valuable time since all the properties, labels, and service plan assignments are copied from the existing subaccount to the new subaccount.

### ⓘ Note

This option is available only on the AWS, Azure, or GCP region.

## Procedure

1. From your global account, navigate to the [Account Explorer](#) page.
2. Do one of the following:
  - Select  [Create](#)  and then choose the source subaccount.
  - Locate the source subaccount in your account hierarchy and then choose [Create as New Subaccount](#) option in its  context menu.
3. Edit the new subaccount's properties and labels, as needed.  
Refer to [Create a Subaccount](#) above to find out more information about the various subaccount properties.
4. **Optional:** If you don't want all the assigned service plans in the selected subaccount to be copied to the new subaccount, turn off the [Copy service plan assignments](#) option.
5. Click [Create](#).

### ⓘ Note

If you selected the option to copy the assigned service plans from the source subaccount to your new subaccount, and any of the required plans are missing in the target location of the new subaccount, or if there is insufficient quota, you'll be informed before the subaccount creation process begins so that you or another global account admin can make the necessary adjustments before you try again.

## Next Steps

After the new subaccount has been created, you can complete its setup, including its authorization configuration, instances, subscriptions, and environment setups.

## Related Information

[Global Accounts \[page 94\]](#)

[Navigate in the Cockpit \[page 2142\]](#)

[Account Model \[page 94\]](#)

## 6.1.1.5.2 Add Members to Your Subaccount

Add members to your subaccount to enable users to access resources available there. Platform users manage subaccounts with cloud management tools, while business users consume applications and services.

### Prerequisites

- You're managing a multi-environment subaccount.  
For more information about adding members to Neo subaccounts, see [Add Members to Your Neo Subaccount](#).
- You must be a subaccount administrator to add other subaccount members.
- The users exist in a trusted identity provider.  
All users of SAP BTP are stored in identity providers, either in the default or in a custom identity provider. SAP BTP needs a copy of the user, sometimes called a shadow user. You assign the shadow user authorizations to access resources in SAP BTP. When a user authenticates, SAP BTP forwards the request to the identity provider.  
For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

### Context

There are two types of users in subaccounts:

- Platform users  
These users are members of your team. Platform users manage the subaccount with the cloud management tools.  
Assign predefined or custom role collections to users who need to manage or view the subaccount in SAP BTP cockpit. Examples of predefined role collections include the following:
  - *Subaccount Administrator*
  - *Subaccount Viewer*For more information about the available role collections, see [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).  
To work in the Cloud Foundry, ABAP, Kyma, and Neo environments, your team members need additional authorizations.  
For more information, see [Giving Access Rights to Platform Users](#) in the *SAP BTP Best Practices Guide*.

#### ⓘ Note

If the subaccount has a Cloud Foundry environment, SAP BTP checks Cloud Foundry users with org or space roles. If there's no corresponding user at the subaccount level, SAP BTP creates one automatically.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

- Business users

These users consume applications and services. Business users don't access cloud management tools like the SAP BTP cockpit. Assign role collections required by the applications and services that the business users consume.

## Procedure

### → Tip

If you're a global account administrator, you can quickly assign yourself as an administrator of a subaccount by going to the [Account Explorer](#) page and then choosing the [Add Me as Admin](#) option in the [Actions \(•••\)](#) context menu of the subaccount. To assign other users, use the following instructions.

This feature isn't available if it has been disabled for your global account.

To disable this feature, send an opt-out request using the component BC-NEO-CIS-OPS. See [Getting Support \[page 3148\]](#) for information about opening a case and getting technical assistance.

1. Navigate to your subaccount.
2. Add a user to your subaccount.  
For more information, see [Create Users \[page 2270\]](#).
3. Assign role collections to the user.  
For more information, see [Assign Users to Role Collections \[page 2278\]](#).

## Results

The user can log on to SAP BTP and access resources according to the role collections you've assigned to the user.

### ⓘ Note

If the new subaccount platform user logs on to SAP BTP cockpit and there's no corresponding platform user at the global account level, SAP BTP creates one automatically.

## Related Information

[User and Member Management \[page 104\]](#)

[Navigate in the Cockpit \[page 2142\]](#)

## 6.1.1.5.3 Enable Environment or Create Environment Instance

Enable an environment or create an environment instance using the SAP BTP cockpit.

### Prerequisites

- You're managing a multi-environment subaccount.
- You're a subaccount administrator.
- You have free quota assigned to your subaccount.

### Context

On a subaccount level, environments constitute the actual platform-as-a-service (PaaS) offering of SAP BTP that allows for the development and administration of business applications.

Each environment comes equipped with specific tools, technologies, and runtimes that you need to build applications. The availability of different environments allows for greater flexibility in your development process.

For more information, see [Environments \[page 47\]](#).

### Procedure

1. Navigate into the subaccount.
2. Choose your method: **Enable Environment or Create an Environment Instance**.
  - **Enable Environment:** On the *Subaccount - Overview* page, click *Enable <Environment>*.
  - **Create an Environment Instance:** See [Environment Instances](#).

### Results

You can use the environment or environment instance in your subaccount.

## 6.1.1.5.4 Change Subaccount Details

Edit subaccounts using the SAP BTP cockpit.

### Context

You edit a subaccount by choosing the relevant action on its tile. It's available in the global account view, which shows all its subaccounts.

The subaccount technical name is a unique identifier of the subaccount on SAP BTP that is generated when the subaccount is created.

You can change to the following subaccount details:

Editable Subaccount Details

Field	Details
Display Name	Specify a human-readable name for your subaccount and change it later on, if necessary. This way you can distinguish between multiple subaccounts.
Description	(Optional) Specify a short descriptive text about the subaccount.
Parent	Change the parent of your subaccount.  When you create a subaccount, this is added as a direct child of the global account you created it in. You can change your subaccount's parent and add it to a directory, move it from one directory to another, or remove from a directory by making the global account its parent once again.
Used for production	(Optional) Select this option if your subaccount is being used for production purposes.  This does not change the configuration of your subaccount. Use this flag for your internal use to operate your production subaccounts in your global account and systems more efficiently. Your cloud operator may also use this flag to take appropriate action when handling incidents related to mission-critical accounts in production systems.  Do not select this option if your account is used for non-production purposes, such as development, testing, and demos.

Field	Details
Labels	<p>(Optional) Assign, change, or remove labels from your subaccount. Labels help to make organizing and filtering your subaccounts easier. For more information, see <a href="#">Labels [page 98]</a>.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>→ Tip</p> <ul style="list-style-type: none"> <li>When adding multiple values to a label, press <code>Enter</code> after each value.</li> <li>When you start typing, any matching label names and values that are already assigned to other subaccounts in your global account are offered as suggestions.</li> </ul> </div>
Enable beta features	<p>(Optional) Enable the subaccount to use services and applications which are occasionally made available by SAP for beta usage on SAP BTP. This option is available to administrators only and is, by default, unselected.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>⚠ Caution</p> <p>You shouldn't use SAP BTP beta features in subaccounts that belong to productive enterprise accounts. For more information, see <a href="#">Important Disclaimers and Legal Information</a>.</p> </div> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>ⓘ Note</p> <p>Once you have enabled this setting in a subaccount you cannot disable it.</p> </div> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p>ⓘ Note</p> <p>If you enable this setting in a subaccount that is already consuming services and subscriptions, these consumed resources will not be automatically updated to beta versions if they exist.</p> </div>

## Procedure

- Choose the subaccount for which you'd like to make changes and choose  on its tile.

You can view more details about the subaccount such as its description and additional attributes by choosing [Show More](#).

- Make your changes and save them.

## Related Information

[Navigate in the Cockpit \[page 2142\]](#)

[Change the Default Database System](#)

### 6.1.1.5.5 View Subaccount Usage Analytics

You can explore, compare, and analyze all your actual usage data for the services and applications that are available in your subaccount.

To monitor and track usage in a subaccount, open the subaccount in the SAP BTP cockpit and choose *Usage Analytics* in the navigation area.

#### Views in the Subaccount 'Usage Analytics' Page

The subaccount *Usage Analytics* page contains views that display usage at different levels of detail:

View	Description
<b>Subaccount</b>	Displays high-level usage information for your subaccount relating to services, business application subscriptions, and Cloud Foundry org spaces.  Some information in this view is displayed only for global account admins.
<b>Services</b>	Displays usage per service plan for the region of the subaccount, and the selected metric and period. Information is shown for all services whose metered consumption in the subaccount is greater than zero.
<b>Spaces</b>	(Cloud Foundry environment only) Displays service plan usage per space for the services shown in the <i>Services</i> view.

#### ⓘ Note

The information displayed in this page depends on the environments that are enabled for your subaccount. For example, information about spaces is displayed only for subaccounts in the Cloud Foundry environment.

Usage values are updated every 24 hours.

Usage data displayed in this cockpit page refers to actual usage, not billed usage.

#### → Tip

- If your subaccount is in a global account that uses the consumption-based commercial model, you can view information about your billed usage for billable services in your global account's *Usage Analytics* page. The global account's *Usage Analytics* page also provides information about cloud-credit usage. A

directory's *Usage Analytics* page also allows you to view usage information for each subaccount that is located in the directory.

- If you use directories to group your subaccounts, you can view usage information by directory in your global account's *Usage Analytics* page or directly in the directory's *Usage Analytics* page.

For more information, see [Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#) and [View Directory Usage Analytics \[page 2171\]](#).

## Working with the Tables and Charts

In the *Services* and *Spaces* views, the usage information is displayed in both tabular and chart formats.

- The tables present accumulated usage values based on the aggregation logic of each service plan and its metric over the selected time period. The usage values are broken down by resource.
- The charts present usage values for one or more service plans or spaces that you select in the adjacent tables. The resolution of the charts is automatically set to days, weeks, or months, depending on the range of the selected time period.

You can perform various actions within the tables and charts:

- In the *Services* and *Spaces* views, select a table row to display the usage information in the chart. You can also select multiple rows to compare usage in the following ways:
  - In the *Services* view, you can compare usage between service plans in the same service. Multi-row selection in the table is possible only when you have selected a single service in the *Service* filter and the row items share the same metric.
  - In the *Spaces* view, you can compare usage between spaces for the same service and service plan. Multi-row selection in the table is possible only when you have selected a single service plan in the *Service Plan* filter.
- In the charts, you can view the data as a column chart or as a line chart.
- To display a larger view of a chart, choose the  *(Zoom)* button.
- Some rounding or shortening is applied to large values. You can mouse over values in the table to view the exact values as tooltips.

## Using the Filters

Use the filters in the *Services* and *Spaces* views to choose which usage information to display.

The *Spaces* view inherits the filter settings, except for the *Period*, from the *Services* view above it. In other words, when you modify filters in the *Services* view, it affects the information that is displayed in the *Spaces* view. You can apply the *Period* filter independently to each view.

In the *Services* view, you can apply the *Metric* filter only when you have selected a service with more than one metric.

Click the  *(Reset)* icon in each view to reset the filters to their default settings. If you reset the filters in the *Services* view, the filters in the *Spaces* are also reset.

## Related Information

[Org Administration Using the Cockpit \[page 2433\]](#)

[Monitoring Usage and Consumption Costs in Your Global Account \[page 2150\]](#)

[What Is the Consumption-Based Commercial Model? \[page 87\]](#)

[View Directory Usage Analytics \[page 2171\]](#)

[Monitoring Usage Information Using APIs of the SAP Usage Data Management Service \[page 2409\]](#)

### 6.1.1.5.6 Delete a Subaccount

Delete subaccounts using the SAP BTP cockpit to clean up your account hierarchy, free up quota used by services in your subaccounts, and to reduce overall costs.

## Delete a Subaccount

### Prerequisites

- You're a global account administrator.

### Procedure

1. In the cockpit, go to the [Account Explorer](#) page of your global account.
2. In the account hierarchy, locate the subaccount that you want to delete.
3. In the [Actions \(•••\)](#) context menu of the subaccount, choose [Delete](#).

The cockpit first checks if the subaccount contains data, such as active subscriptions, service instances, brokers, or platforms.

4. If the subaccount is empty, then confirm the operation, after which the subaccount is deleted.

If you are a subaccount administrator and the subaccount contains data, such as applications, service instances, spaces, active subscriptions, brokers, and members, then we strongly recommended that you visit the [Instances and Subscriptions](#) page to review the content in the subaccount and to manually remove the content from there. Then you can come back and safely delete the subaccount.

#### Note

If you are not a subaccount administrator, then you won't be able to access the [Instances and Subscriptions](#). Instead, you'll have to ask a subaccount administrator in your organization to review the content in this subaccount and to manually remove the content from there.

### Caution

If the subaccount contains data, then you're also given the option to use the **force-delete** option, which deletes the subaccount and all its data for you.

You must use this option with extreme caution because any data in the subaccount is deleted permanently, including productive data, without any recovery option.

It is your responsibility to confirm that the subaccount is safe to delete before using the force-delete option. If you cannot check the content of the subaccount, you can get the subaccount reviewed by other qualified members of your organization.

### Note

The deletion may take up to 12 hours to complete depending on the amount of content in your subaccount. If your subaccount contains service instances, subscriptions, and runtime environments, these need to be deprovisioned as part of the purge operation, which is a timely process. After the subaccount is deleted, it could take a few more days for some related services to be deleted. Note that you won't be charged for any continued usage of these services in the subaccount during the deletion cleanup.

## Related Information

[Navigate in the Cockpit \[page 2142\]](#)

[Relationship Between Global Accounts, Subaccounts, and Directories \[page 98\]](#)

### 6.1.1.6 Managing Entitlements and Quotas Using the Cockpit

When you purchase an enterprise account, you are entitled to use a specific set of resources, such as the amount of memory that can be allocated to your applications.

An entitlement equals your right to provision and consume a resource. A quota represents the numeric quantity that defines the maximum allowed consumption of that resource.

Entitlements are managed at the global account level, where services plans and their allowed quota are distributed to subaccounts, and then consumed by the subaccounts. When quota is freed at the subaccount level, it becomes available again at the global account level.

### Note

Only global account administrators can configure entitlements and quotas for subaccounts.

There are two places in the SAP BTP cockpit where you can view entitlements and assign service plans and quotas for subaccounts - at the global account level and at the subaccount level. Depending on your permissions, you may only have access to one of these pages. You can find more details below:

#### Entitlements Pages

Page in cockpit	Navigation Level	Visible to	Permissions
▶ <a href="#">Entitlements</a> ▶ <a href="#">Entity Assignments</a> ▶	Global account level	Global account administrators only	<ul style="list-style-type: none"> <li>• View &amp; Edit - Global account administrators</li> </ul>
▶ <a href="#">Entitlements</a> ▶ <a href="#">Service Assignments</a> ▶	Global account level	Global account administrators only	<ul style="list-style-type: none"> <li>• View - Global account administrators (you cannot make changes to entitlements or quota assignments from this page)</li> </ul>
<a href="#">Entitlements</a>	Subaccount level	Subaccount members (regardless of whether they are also global account administrators or not)	<ul style="list-style-type: none"> <li>• View - Subaccount members who are not global account administrators</li> <li>• View &amp; Edit - Global account administrators who are also subaccount members</li> </ul>

#### ⓘ Note

You can also assign service plans to directories. These directories must be configured to manage entitlements. See [Configure Entitlements and Quotas for Directories \[page 2188\]](#).

To assign plans to managed directories you must a global account administrator.

Similarly, if you're assigning plans to a subaccount that is under a directory that has the user management feature enabled, then you must be the directory administrator or a global account administrator.

The ▶ [Entitlements](#) ▶ [Entity Assignments](#) ▶ is where you view and edit assignments and quota from the perspective of subaccounts and directories. This page contains these views (tabs):

View	Description
<a href="#">Manage Assignments</a>	Allows global account admins to distribute the entitled service plans of their global account to subaccounts and directories.
<a href="#">View Commercial Information</a>	Allows global account admins and viewers to view the commercial source of their service entitlements. This feature is not available for trial accounts.

For information about setting assignments and quota, see:

- [Configure Entitlements and Quotas for Directories \[page 2188\]](#)
- [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

The ▶ [Entitlements](#) ▶ [Service Assignments](#) ▶ is where you view assignments and quota from the perspective of services. This page contains these views (tabs):

View	Description
All Services	<p>Provides an overview of all available service entitlements of your global account.</p> <p><b>→ Tip</b></p> <p>The <i>Assigned To</i> column in this view shows you the total number of directories and subaccounts to which each service plan is assigned in your global account. You can click on the links in this column to quickly navigate to the selected service in the <i>Assignments by Service</i> view and display more details about these assignments.</p> <p>Turn on the <i>Show commercial info</i> option to display additional columns in the main table. These columns can help you to gain a better understanding of the relationship between contractual service entitlements and the technical assets (services and plans) in your global account.</p>
Assignments by Service	<p>Allows you to choose specific services and see how their service plans are assigned to subaccounts and directories within your global account.</p> <p><b>→ Tip</b></p> <p>Turn on the <i>Show commercial info</i> option to display additional columns in the main table. These columns can help you to gain a better understanding of the relationship between contractual service entitlements and the technical assets (services and plans) in your global account.</p>

## ⓘ Note

- To subscribe to a multitenant application, you must first assign its plan to the specific subaccount (in one of the *Entitlements* pages). If you remove a subaccount's entitlement to a multitenant application, any subscriptions to it from that subaccount will stop working.
- Before a subaccount admin can enable a quota-based environment, such as Kyma, the subaccount admin must first assign the environment's plan to the subaccount (in one of the *Entitlements* pages). Other non-quota-based environments, such as Cloud Foundry, are available by default to all subaccounts, and therefore are not available as entitlements.

In the Cloud Foundry environment, you can further distribute the quotas that are allocated to a subaccount. This is done by creating space quota plans and assigning them to the spaces. For more information on space quota plans in the Cloud Foundry environment, see:

- [Managing Space Quotas \[page 2447\]](#)
- <https://docs.cloudfoundry.org/adminguide/quota-plans.html> ↗

**→ Tip**

Optionally, you can configure assignments and quotas for supported services that you own or subscribe to from supported non-SAP cloud vendors. These services can be consumed in SAP BTP in your subaccounts alongside your self-developed services and those provided by SAP. This functionality requires first setting up a resource provider instance for your provider account in the cockpit. For more information, see [Managing Resource Providers \[page 2147\]](#).

#### Note

- In global accounts that use the consumption-based commercial model, SAP BTP, Cloud Foundry runtime is not listed in the [Entitlements](#) pages in the SAP BTP cockpit. A technical limit of 200 GB of Cloud Foundry runtime memory is assigned by default to every subaccount. The limit defines the maximum amount of runtime memory that can be used in the subaccount. Note that the "Standard" plan in the consumption-based commercial model is a paid plan and that you are billed based on the amount of Cloud Foundry runtime you consume. For more information on consumption monitoring, see [Consumption Monitoring](#).
- If you need to increase this limit, report an incident to [SAP support](#) on the BC-NEO-CIS component. This also applies to other services that have a technical quota limit.

## Related Information

[Entitlements and Quotas \[page 100\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

[Configure Entitlements and Quotas for Directories \[page 2188\]](#)

[Create Space Quotas \[page 2449\]](#)

[Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#)

[Tutorial: Manage Entitlements on SAP BTP Trial](#)

[Troubleshooting: Entitlement and Quota Problems](#)

### 6.1.1.6.1 Configure Entitlements and Quotas for Directories

Distribute entitlements that are available in your global account to directories by adding service plans and their allowed quotas by using SAP BTP cockpit.

## Enable Entitlement Management for a Directory

Before you can assign plans and quotas to a directory so that they can be further distributed to the subaccounts in the directory, you must first enable the **entitlement management** feature for the directory. This configuration only needs to be done once for each directory. In this documentation, we'll refer to these types of directories as "managed directories".

#### Note

- In the account hierarchy, each directory path can have only one directory enabled with the entitlement management feature (or the user management feature). Consequently, if a directory in the path already has one of these features enabled, it is not possible to enable either of them for other parent or child directories within the same path as this directory.

- Enabling entitlement management for a directory that already contains subaccounts will result in the automatic assignment of any existing assignments and quotas (for the subaccounts) to the directory.

## Prerequisites

- You are a global account administrator.

## Procedure

- There are several ways you can enable the entitlement management feature for a directory:
- In the *Account Explorer*, click the **Actions** button for a specific directory and select *Edit*. In the *Edit Directory* dialog box, under *Advanced*, choose the *Enable entitlement management* option.
- Navigate to the directory's *Entitlements* page and select *Enable Entitlement Management*.
- Navigate to your global account and open the **Entitlements > Entity Assignments** page from the navigation panel. Select the  icon in the *Select Entity* dropdown menu.

Then, in the *Select Entities* dialog box, choose the directory for which you want to manage assignments. Click *Select* and then *Enable Entitlement Management*.

### → Tip

To disable entitlement management for a directory, redo these steps from the global account's **Entitlements > Entity Assignments** page, but choose *Disable Entitlement Management* at the end. When you disable entitlement management, any assignments and quota that are currently assigned to the directory are "returned" to the global account and are then available for distribution to other directories and subaccounts.

## Next Steps

Once you've enabled entitlement management for a directory, you can now configure its service plan and quota assignments as described below.

# Configure Assignments and Quotas for Managed Directories

Use this procedure to configure the service plan assignments and quotas for one or more managed directories.

### ⓘ Note

To get an overview of all the services and plans available in your global account, navigate to the **Entitlements > Service Assignments** page. There you can see the global usage of each service plan.

in the [All Services](#) view, as well as the detailed assignments of each service across subaccounts and directories in the [Assignments by Service](#) view.

- In the [All Services](#) view of this page, the [Assigned To](#) column shows you the total number of directories and subaccounts to which each service plan is assigned in your global account. You can click on the links in this column to quickly navigate to the selected service in the [Assignments by Service](#) view and display more details about these assignments.
- You can turn on the [Show commercial info](#) toggle option in the [Service Assignments](#) page to display additional table columns that can help you to gain a better understanding of the relationship between contractual service entitlements and the technical assets (services and plans) in your global account; in other words, how each entitled plan was added to your global account.
- The [Service Assignments](#) page is for viewing purposes only; you cannot make any assignment changes from it.

You can also access the [Entitlements](#) [Service Assignments](#) page from the directory level (only for directories that are configured to manage entitlements) to display the directory's assignments.

## Prerequisites

- You are a global account administrator.  
Note that if the directory also has the user management feature enabled, then you must be either the directory administrator or a global administrator. See [Manage Users in Directories \[page 2169\]](#).
- The service and applications that you intend to assign to your directories must be entitled to your global account.
  - For enterprise accounts that use the subscription-based commercial model, your account receives only the services that you've already purchased and subscribed to according to your SAP BTP contract. To access additional services, at an extra cost, you can modify your contract via your sales representative or account executive.
  - For enterprise accounts that use the consumption-based commercial model, your global account receives all pay-for-use and free services that are eligible for this commercial model.
  - If you have a trial account, you'll receive a restricted set of free platform resources and services for a limited time period. To access additional services, you'll have to switch to an enterprise account.

## Procedure

1. Navigate to your global account.
2. Choose [Entitlements](#) [Entity Assignments](#) from the left hand-side navigation.  
Alternatively, you can navigate to a directory from the [Account Explorer](#), and then access the [Entitlements](#) [Entity Assignments](#) page for that specific directory.
3. Open the [Manage Assignments](#) view (tab).

Skip this step if you have a trial account.

### → Tip

You can use the *View Commercial Information* view in this page to display comprehensive commercial information about the entitlements in your global account (you cannot make assignments in this view). Note that this view is not available for trial accounts.

4. Select the  icon in the *Directories / Subaccounts* dropdown menu.
5. In the *Select Subaccounts and Directories* dialog box, choose the directories for which you want to manage assignments.
6. Click *Select*.

### ⓘ Note

If you select a directory for which entitlement management is not enabled, you can click *Enable Entitlement Management* to enable the feature. Remember that you can enable this feature only for a single directory in a path.

You get a table for each of the directories you selected, displaying the current assignments and quotas. You can choose *Show subaccount assignments* next to the table title to see the assignments and quotas for the subaccounts in a specific directory.

### → Tip

Under the *Service Technical Name* column, click on the  (*Service Details*) icon of any service to access its service-related links to documentation, support information, and SAP Discovery Center.

7. Choose *Edit* to start editing assignments for a particular directory.

### ⓘ Note

You can only edit assignments for one directory at a time.

8. You can now edit the assignments table:

Action	Steps
<b>Add new service plans to the directory</b>	<p>Choose <i>Add Service Plans</i> and from the dialog box select the services and the plans from each service that you would like to add to the directory. Choose <i>Add &lt;x&gt; Service Plans</i> in the dialog box to confirm.</p> <p>Remember, the services that you see depend on the type of global account you have and your contract details (see the prerequisites above for more information).</p> <p>Once you've added new service plans, you can also change the quota for those plans as described in the following row and enable the option to auto-assign quota to new subaccounts that are added to the directory. For service plans where quota can be increased or decreased, you can also specify what amount should be auto-assigned to each subaccount that is created or added to the directory.</p>

### ⓘ Note

To subscribe a subaccount to a multitenant application, you must first assign the application to the specific subaccount using the process described in [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#).

Action	Steps
	<p>→ <b>Tip</b></p> <p>If your global account is configured to consume remote services from a non-SAP cloud vendor (resource provider), an additional dropdown list is displayed in the <i>Service Details</i> pane, where you can choose a configured resource provider. You can then choose the required service plans that are available for the selected resource provider to add them to the subaccount.</p> <p>For more information, see <a href="#">Managing Resource Providers [page 2147]</a>.</p>
<b>Edit the quota for one or more service plans</b>	<p>Use <b>+</b> (<i>Plus</i>) and <b>-</b> (<i>Minus</i>) in the <i>Quota Assignment</i> column to increase or decrease the quota for each service plan.</p> <p>Note some service plans do not have a numeric quota assignment. In other words, you do not assign a specific quantity but grant access to the plan by simply assigning it to the subaccount. See <a href="#">Quotas in Managing Entitlements and Quotas Using the Cockpit [page 2185]</a>.</p>
	<p>→ <b>Tip</b></p> <p>If you would like to have quota automatically assigned to subaccounts that are created or moved to the directory, select the <i>Auto-Assign to Subaccounts</i> checkbox and follow the same process to set the amount that should be auto-assigned to each subaccount.</p> <ul style="list-style-type: none"> <li>The auto-assign feature doesn't apply to subaccounts that are already in the directory when the feature is enabled.</li> <li>You cannot use the auto-assign feature with beta services and applications.</li> </ul>
<b>Set a quota limit to unlimited service plans that use the consumption-based commercial model</b>	<p>The usage of most service plans is unlimited by default in global accounts that use the consumption-based commercial model. To control costs of such plans, you can set a quota limit on the service plans for this directory by first selecting the checkbox in the <i>Set Quota Limit</i> column.</p> <p>Now, you can increase or decrease the quota for this service plan in the <i>Quota Assignment</i> column by using <b>+</b> (<i>Plus</i>) and <b>-</b> (<i>Minus</i>).</p> <p>Note some service plans do not have a numeric quota assignment. In other words, you do not assign a specific quantity but grant access to the plan by simply assigning it to the directory (and then subsequently to the subaccount). See <a href="#">Quotas in Managing Entitlements and Quotas Using the Cockpit [page 2185]</a>.</p>
<b>Delete a service plan and its quota from the directory</b>	Choose  ( <i>Remove Service Plan</i> ) from the <i>Actions</i> column.

- Once you're done, choose **Save** to save the changes and exit edit mode for that directory.
- Optional:** Repeat steps 7 to 9 to configure assignments for the other selected directories.

## Related Information

[Entitlements and Quotas \[page 100\]](#)

[Directories \[page 96\]](#)

[Manage Users in Directories \[page 2169\]](#)

[Create a Directory \[page 2166\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

[Using Free Service Plans \[page 91\]](#)

[Trial Accounts and Free Tier \[page 82\]](#)

## 6.1.1.6.2 Configure Entitlements and Quotas for Subaccounts

Distribute the entitlements that are available in your global account by adding service plans and their allowed quotas to your subaccounts using SAP BTP cockpit.

### Prerequisites

- You must be a global account administrator to configure subaccount assignments. Note that if the subaccount is under a directory that is configured to manage users, then you must be either the directory administrator or a global account administrator to configure the subaccount's assignments.
- The service and applications that you intend to assign to your subaccounts must be entitled with the required quota to your global account. If you're assigning a service plan to a subaccount, which is under a directory that is configured to manage entitlements, then the managed directory must have the relevant service plans and needed quota assigned to it so that you can distribute them to the directory's subaccounts, as required.
  - For enterprise accounts that use the subscription-based commercial model, your account receives only the services that you've already purchased and subscribed to according to your SAP BTP contract. To access additional services, at an extra cost, you can modify your contract via your sales representative or account executive.
  - For enterprise accounts that use the consumption-based commercial model, your global account receives all pay-for-use and free services that are eligible for this commercial model.
  - If you have a trial account, you'll receive a restricted set of free platform resources and services for a limited time period. To access additional services, you'll have to switch to an enterprise account.
- To assign beta service plans, your subaccount must have the *Enable beta features* option selected. See [Change Subaccount Details \[page 2180\]](#). Only applies to non-production subaccounts.

## Context

You can distribute assignments and quotas across subaccounts within a global account from these places in the cockpit:

- The  [Entitlements](#)  page at the global account level (visible only to global account administrators).
- The  [Entitlements](#)  page at the directory level by first navigating into the directory from the [Account Explorer](#). This applies only subaccounts that resides in the path of a directory that is configured to manage entitlements.
- The [Entitlements](#) page at the subaccount level (visible to all subaccount members, but editable only by global account administrators) by first navigating into the subaccount from the [Account Explorer](#).

For more information, see [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

### → Tip

To get an overview of all the services and plans available in your global account, navigate to the  [Entitlements](#)  page. There you can see the global usage of each service plan in the [All Services](#) view, as well as the detailed assignments of each service across subaccounts and directories in the [Assignments by Service](#) view.

- The [Service Assignments](#) page is for viewing purposes only; you cannot make any assignment changes from it.
- In the [All Services](#) view of this page, the [Assigned To](#) column shows you the total number of directories and subaccounts to which each service plan is assigned in your global account. You can click on the links in this column to quickly navigate to the selected service in the [Assignments by Service](#) view and display more details about these assignments.
- You can turn on the [Show commercial info](#) toggle option in this page to displays additional table columns that can help you to gain a better understanding of the relationship between contractual service entitlements and the technical assets (services and plans) in your global account; in other words, how each entitled plan was added to your global account.

If you navigate to a subaccount from the [Account Explorer](#), you view its assignments in its [Overview](#) page view.

### ⓘ Note

- To subscribe to a multitenant application, you must first assign it to the specific subaccount using the same process explained below. If you remove a subaccount's assignments to a multitenant application, any subscriptions to it from that subaccount stop working.
- Before a subaccount admin can enable a quota-based environment, such as Kyma, the subaccount admin must first assign the environment's plan and quota to the subaccount. Other non-quota-based environments, such as Cloud Foundry, are available by default to all subaccounts, and therefore aren't available as entitlements.

# Configure Assignments and Quotas from Your Global Account

## Procedure

1. Navigate to your global account.
2. Choose *Entitlements* *Entity Assignments* from the left hand-side navigation.

### → Remember

There are other ways you can access the subaccount to manage its assignments, such as navigating to the subaccount or its managed directory from the *Account Explorer*. See above for more information.

3. Open the *Manage Assignments* view (tab).

Skip this step if you have a trial account.

### → Tip

You can use the *View Commercial Information* view in this page to display comprehensive commercial information about the entitlements in your global account (you cannot make assignments in this view). Note that this view is not available for trial accounts.

4. Select the icon in the *Directories / Subaccounts* dropdown menu.
5. In the *Select Subaccounts and Directories* dialog box, select all the subaccounts that you want to manage assignments.
6. Choose *Select* to apply the filter.

You get a table for each of the subaccounts you selected, displaying the current assignments and quota.

### → Tip

Under the *Service Technical Name* column, click on the (*Service Details*) icon of any service to access its service-related links to documentation, support information, and SAP Discovery Center.

7. Choose *Edit* to start editing assignments for a particular subaccount.

### ⓘ Note

You can only edit assignments for one subaccount at a time.

8. You can now edit the assignments table:

Action	Steps
<b>Add new service plans to the subaccount</b>	<p>Choose <i>Add Service Plans</i> and from the dialog box, select the services and the service plans from each service that you would like to add to the subaccount. Choose <i>Add &lt;x&gt; Service Plans</i> in the dialog box to confirm.</p> <p>Remember, the services that you see depend on the type of global account you have and your contract details (see the prerequisites above for more information).</p>

Action	Steps
	<p>→ Tip</p> <p>If your global account is configured to consume remote services from a non-SAP cloud vendor (resource provider), an additional dropdown list is displayed in the <i>Service Details</i> pane, where you can choose a configured resource provider. You can then choose the required service plans that are available for the selected resource provider to add them to the subaccount.</p> <p>For more information, see <a href="#">Managing Resource Providers [page 2147]</a>.</p>
Edit the quota for one or more assigned service plans	<p>Use <b>+</b> (<i>Plus</i>) and <b>-</b> (<i>Minus</i>) in the <i>Quota Assignment</i> column to increase or decrease the quota for each service plan.</p> <p>Note some service plans do not have a numeric quota assignment. In other words, you do not assign a specific quantity but grant access to the plan by simply assigning it to the subaccount. See Quotas in <a href="#">Managing Entitlements and Quotas Using the Cockpit [page 2185]</a>.</p>
Set a quota limit to unlimited service plans that use the consumption-based commercial model	<p>The usage of most service plans is unlimited by default in global accounts that use the consumption-based commercial model. To control costs of such plans, you can set a quota limit on the service plans for this subaccount by first selecting the checkbox in the <i>Set Quota Limit</i> column.</p> <p>Now, you can increase or decrease the quota for this service plan in the <i>Quota Assignment</i> column by using <b>+</b> (<i>Plus</i>) and <b>-</b> (<i>Minus</i>).</p>
Delete a service plan and its quota from the subaccount	<p>Choose  (<i>Remove Service Plan</i>) from the <i>Actions</i> column.</p>

9. Once you're done, choose **Save** to save the changes and exit edit mode for that subaccount.
10. **Optional:** Repeat steps 7 to 9 to configure assignments for the other selected subaccounts.

## Related Information

[Entitlements and Quotas \[page 100\]](#)

# Configure Assignments and Quotas from a Subaccount

If you're working in a subaccount and realize you're missing assignments or quota, you can edit the assignments directly from that subaccount.

## Prerequisites

In addition to being a global account administrator, you must also be a member of the subaccount to be able to access that subaccount.

## Procedure

1. Navigate into the subaccount where you would like to configure the assignments.
2. Choose *Entitlements* from the left hand-side navigation to see the assignments for your subaccount.
3. Choose *Edit*.
4. You can now edit the assignments table as described in the procedure above.
5. Once you're satisfied with the changes, choose *Save* to save and exit edit mode.

## Related Information

[Entitlements and Quotas \[page 100\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

[Configure Entitlements and Quotas for Directories \[page 2188\]](#)

[Using Free Service Plans \[page 91\]](#)

[Trial Accounts and Free Tier \[page 82\]](#)

## 6.1.1.7 Subscribe to Multitenant Applications Using the Cockpit

Subscribe to multitenant applications from the [Subscriptions](#) page in the SAP BTP cockpit.

### Prerequisites

- If you're subscribing to an application provided by SAP:
  - If your global account uses the subscription-based commercial model, then you must have purchased SaaS licenses for the applications you want to consume. See <https://cloudplatform.sap.com/pricing.html>. You can also contact us on [SAP BTP](#) or via an SAP sales representative.

#### ⓘ Note

If your global account is licensed with the consumption-based commercial model, then you are eligible to subscribe to any multitenant application that is available to this model.

- You have created a multi-environment subaccount. See [Create a Subaccount \[page 2173\]](#).
- If you are an application owner who is subscribing consumer tenants to your own multitenant application that you created in the Cloud Foundry environment:
  - The application must be deployed to your provider subaccount, configured, and registered as described in [Developing Multitenant Applications in the Cloud Foundry Environment \[page 359\]](#).
  - You have created a multi-environment subaccount for each application consumer in the region in which the application is deployed. See [Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 377\]](#).
- You are an administrator of the subaccount.
- Your subaccount is entitled to the multitenant application. See [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

### Context

The instructions provided here apply whether you are an SAP customer subscribing to an application provided by SAP or you are an application owner who is sharing your multitenant application with your consumers.

### Procedure

1. Open your global account in the cockpit.
2. Do one of the following:
  - If you are subscribing to an SAP application, open your subaccount.
  - If you are sharing your multitenant application with other consumers in your global account, open the subaccount of each consumer.

See [Navigate in the Cockpit \[page 2142\]](#).

3. In the navigation area of each subaccount, choose .

All resources you are entitled to consume in the subaccount appear as tiles.

#### Note

To create a subscription, you need to open the *Create* wizard. There are 2 ways to reach this wizard, either from the *Service Marketplace* or from the *Instances and Subscriptions* page.

In steps 3, 4, 5, and 6, we describe how to reach the wizard from the *Service Marketplace* page.

This way is useful if you want to get an overview of all the applications your subaccount is entitled to use and if you want to get more details about the available plans before you create a subscription.

If you already know these details, select , click on *Create* in the top-right corner, and skip to step 7 to learn about the wizard.

4. In the filters area, select the *All Types* filter, and from the dropdown menu, select *Application*.

You can use other filters to narrow down your search by the environments for which the application is available, capabilities, and its status.

*Active* status means you are currently subscribed to that application.

5. Click the application name to open its *Overview* page.

There, you can find more information about the application including all the available plans.

#### Tip

You can also create a new subscription without opening its overview by selecting the three dots in the top-right corner of the tile.

6. Choose *Create*.
7. In the wizard that opens, select one of the plans of type subscription and choose *Create*.

#### Note

The wizard contains more than one step if additional subscription configuration input parameters are provided by the app provider.

A confirmation popup appears with the link to the *Instances and Subscriptions* page, where you can follow the status of your new subscription under the *Subscriptions* table.

The [Go to Application](#) link becomes available once the subscription is activated. Choose the link to launch the application and obtain its URL.

To find the link, either click on the application's name or click on the ([Go to Application](#)) icon in the *Application* column.

#### Tip

To update an existing subscription plan, select the three dots at the end of the subscription row, and from the menu, select *Update*.

### Note

You can update a subscription plan only if additional plans for the subscription are entitled to the subaccount you're using and if your subscription is eligible for a plan update.

### Tip

To remove an existing subscription, select the three dots at the end of the subscription row, and from the menu, select *Delete*. All data related to the multitenant application will be deleted in the respective subaccount.

8. **Optional:** Assign labels to the subscription to make it searchable and organized within the subaccount according to various criteria.

- a. In the navigation area, choose  *Services*  *Instances and Subscriptions* .
- b. Find the subscription to which you want to assign labels under the *Subscriptions* section of the page.
- c. Select the actions (•••) menu and from the dropdown list, choose *Add Labels*.

### Note

If there are already labels assigned to the subscription, the action becomes *Change Labels*.

You can see the existing labels under the *Labels* column.

If you do not see the column, unhide it by clicking on  (*Configure Table Columns*).

For more information, see [Labels \[page 98\]](#).

## Next Steps

1. Configure user access to the application.  
See [Configure Application Roles and Assign Roles to Users \[page 2201\]](#).
2. If the application consumer isn't the subaccount owner or administrator, provide the consumer-specific URL of the application to the consumer or the users of the application.

## Related Information

[Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

## 6.1.1.7.1 Configure Application Roles and Assign Roles to Users

View, create, and modify application roles and then assign users to these roles using the SAP BTP cockpit.

### Prerequisites

Your subaccount is subscribed to a multitenant application in the Cloud Foundry environment. See [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#).

### Context

You can use any SAML 2.0 standard compliant identity provider. See [Trust and Federation with Identity Providers \[page 2204\]](#).

## View, Create, and Modify Application Roles

### Procedure

1. Navigate to your subaccount. For more information, see [Navigate in the Cockpit \[page 2142\]](#).
2. In the navigation area, choose [Services](#) [Instances and Subscriptions](#) .
3. Under the *Subscriptions* table, click on the row with the application for which you want to manage app roles to open its *Overview* page.
4. Open the tab *Roles* to view, create, and modify the application roles.

#### → Tip

You can also manage roles without opening the application overview page, by selecting the three dots at the end of the application row. Choose *Manage Roles* from the drop-down menu.

## Assign Users to Application Roles

### Procedure

1. Navigate to your subaccount. For more information, see [Navigate in the Cockpit \[page 2142\]](#).
2. Choose [Security](#) [Roles Collections](#) in the navigation area and include the roles into the roles collection.

3. Choose   and assign role collections to users.

## Related Information

[User and Member Management \[page 104\]](#)

### 6.1.1.8 Security Administration: Managing Authentication and Authorization

This section describes the tasks of administrators of SAP BTP. Administrators ensure user authentication and assign authorization information to users and user groups.

Since identity providers provide the users or user groups, you make then sure that there is a trust relationship between your subaccount and the identity provider. This is a prerequisite for authentication. You can manage the authorizations of the users.

#### Authentication

A user account corresponds to a particular user in an identity provider. The user is always authenticated using an external identity provider. We recommend to use a custom tenant of Cloud Identity Services. You can connect Cloud Identity Services to your corporate identity provider.

SAP BTP distinguishes two types of users. Platform users are usually administrators, operator, or developers. They have full access and give permissions at global account, directory, and subaccount level. Business users use the applications deployed to SAP BTP. They are, for example, end users of SaaS apps or of custom applications.

For more information, see the related links.

#### Authorization

Application developers create and deploy application-based authorization artifacts for business users. Administrators use this model to manage roles, build role collections, and assign these collections to users or user groups. In this way, they control the users' permissions.

To perform the functions related to authorization artifacts, account administrators can have multiple options. Here are some of the options:

- The SAP BTP cockpit covers all authorization functions. Its user interface offers easy-to-use and clear navigation.

- There is also a command line option to manage most authorization artifacts. If you prefer working in a terminal or automating operations, use the SAP BTP command line interface (btp CLI). It's suitable for repetitive tasks.
- Especially if you need to perform bulk operations or programmatically access the authorizations, we recommend to use the REST API for authorizations of the SAP Authorization and Trust Management service.
- Administrators can also use the Terraform Provider for SAP BTP within Infrastructure as Code to manage some of the authorization functions.

You find the all available options and tools for managing authorizations in the account administration overview. See [Account Administration \[page 2139\]](#).

Setting Up Authorization Artifacts (Account Administrators)

Task	Links
Assign the role collection to the users provided by an identity provider	<a href="#">Working with Role Collections [page 2274]</a>
(If you do use a custom identity provider) Assign the role collections to user groups	<a href="#">Map Role Collections to User Groups [page 2281]</a>
Assign the role collections to users and user groups, manage attribute mappings	<a href="#">Mapping Role Collections in the Subaccount [page 2281]</a>
Create a role collection and assign roles to it	<a href="#">Define a Role Collection [page 2275]</a>
Use an existing role or create a new one using role templates	<a href="#">Add Roles to Role Collections on the Application Level [page 2293]</a>

### ⓘ Note

When users log on, their authorizations are stored in each user's current session. These authorizations are not dynamically updated and are removed from there only when the session is terminated. This means that, after changes of role collection assignments of a user, these changes only become effective after the user logged out and logged on again.

## Related Information

- [Trust and Federation with Identity Providers \[page 2204\]](#)
- [Monitoring and Troubleshooting \[page 3038\]](#)
- [SAP Authorization and Trust Management Service \[page 3023\]](#)
- [Default Identity Provider \[page 2258\]](#)
- [User and Member Management \[page 104\]](#)

## 6.1.1.8.1 Trust and Federation with Identity Providers

SAP BTP supports identity federation, a concept of linking and reusing digital identities of a user base across loosely coupled systems. Identity federation frees applications on When setting up accounts you need to assign users. While we provide you with your first users from the default identity provider to get you started, your organization has identity providers that you want to integrate.SAP BTP as well as the platform itself from the need to obtain and store the credentials of users and to authenticate them. Instead, the user base is reused from identity providers, which support the administration of digital user identities, authentication, and authorizations in a centralized and decoupled manner. To enable communication between SAP BTP and identity providers, you must cross-configure the communication endpoints of the involved systems, establishing a trust relationship between them.

### → Recommendation

We recommend that you use a custom tenant of SAP Cloud Identity ServicesWhen setting up accounts you need to assign users. While we provide you with your first users from the default identity as identity provider and connect a potential corporate identity provider there. For platform users, the use of SAP Cloud Identity Services is mandatory. If you don't have a tenant yet, check [Getting a Tenant](#).

To connect your corporate identity provider to SAP Cloud Identity Services, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication](#) and [SAP Cloud Identity Services](#)

### **Identity Provider and the SAP Authorization and Trust Management Service in SAP BTP Architecture**

SAP Cloud Identity Services is a multitenancy-enabled identity provider for all SAP cloud applications and optionally on-premise applications. The service provides capabilities for authentication, single sign-on, authorizations, identity lifecycle management, and on-premise integration as well as self-services like self-registration or password reset.

SAP has its own SAP Cloud Identity Services tenant, SAP ID service. SAP ID service is the default identity provider of SAP BTP and where you register to get initial access to SAP BTP. Trust to SAP ID service is preconfigured by default.

We recommend that you request your own SAP Cloud Identity Services tenant (see [Getting a Tenant](#)). To establish trust with your identity provider, proceed as follows.

For business users: [Business Users \[page 2205\]](#)

For platform users: [Platform Users \[page 2238\]](#)

## Related Information

[Settings for Default SAML Federation Attributes of Identity Providers for Business Users \[page 2231\]](#)

## 6.1.1.8.1.1 Business Users

**User accounts** enable users to log on to SAP BTP, access subaccounts, and to use applications according to the permissions granted to them. In this context, it's important to understand the difference between the two types of users that we refer to: business users and platform users.

**Business users** use the applications that are deployed to SAP BTP. For example, the end users of SaaS apps or services, such as SAP Build Work Zone, or end users of your custom applications are business users.

In the Cloud Foundry environment, application developers ([platform users \[page 2238\]](#)) create and deploy application-based security artifacts for business users. Administrators use these artifacts to assign roles, build role collections, and then assign those role collections to business users or user groups. The assignment of role collections enables administrators to control the permissions that the users have in the deployed applications.

For business users, there's a [default identity provider \[page 2258\]](#). We expect that you have your own user base. We recommend that you configure the SAP Cloud Identity Services service as the identity provider and connect SAP Cloud Identity Services to your own corporate identity provider.

### Related Information

[Platform Users \[page 2238\]](#)

[Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#)

[Manually Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2224\]](#)

[Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 2228\]](#)

[Using Multiple Identity Providers from the Same Subaccount \[page 2233\]](#)

## 6.1.1.8.1.1.1 Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services

Use your SAP Cloud Identity Services tenant as an identity provider or a proxy to your own identity provider hosting your business users. This method avoids the upload and download of SAML meta data by using OpenID Connect (OIDC) to establish trust.

### Prerequisites

- You've subaccount administrator permissions.
- You've a tenant of SAP Cloud Identity Services.  
For more information, see [Tenant Model and Licensing](#) in the documentation for SAP Cloud Identity Services.

- The SAP Cloud Identity Services tenant is associated with the customer IDs of the relevant global account of SAP BTP.

For more information, see [Reuse SAP Cloud Identity Services Tenants for Different Customer IDs](#) in the documentation for SAP Cloud Identity Services.

## Context

### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Consider the following restrictions and tips before you start.

### ⚠ Restriction

- You can only establish trust with a single tenant of SAP Cloud Identity Services per subaccount using this method.
- Your SAP Cloud Identity Services tenant can be changed only when no SAP Cloud Identity Services-based subscriptions (for example, SAP Build or SAP Integration Suite, advanced event mesh) exist.
- You've already created a trust configuration with a custom identity provider for applications. In this case, you can't add a trust configuration with the same SAP Cloud Identity Services tenant using another protocol.

Consider the upper limits for trust configurations in the subaccount. See [Limits for the Subaccount \[page 3089\]](#).

### → Tip

- We recommend that you always use SAP Cloud Identity Services as single identity provider for SAP BTP. If you use corporate identity providers, connect them to your SAP Cloud Identity Services tenant, which then acts as a hub. We especially recommend this if you are using multiple corporate identity providers. For platform users, the use of SAP Cloud Identity Services is mandatory.  
For more information, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication](#) and [SAP Cloud Identity Services](#)
- We provide APIs so you can perform this procedure programmatically. For more information, see the [Identity Provider Management API](#) on [SAP Business Accelerator Hub](#).

## Procedure

1. In the SAP BTP cockpit, go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose [Security](#) [Trust Configuration](#).
2. Choose [Establish Trust](#).  
The [Configure Tenant](#) wizard opens.
3. Choose the SAP Cloud Identity Services tenant. The identity providers listed are the SAP Cloud Identity Services tenants associated with your customer ID. Continue with [Next](#).

4. Choose the domain configured with the SAP Cloud Identity Services tenant and continue with [Next](#).
5. You can change the name and the description of the tenant, display and change the origin key, and provide a link text for user logon (see [Using Multiple Identity Providers from the Same Subaccount \[page 2233\]](#)). Continue with [Next](#).
6. Review your configuration and confirm using [Finish](#).

## Results

You've configured trust in your tenant of the SAP Cloud Identity Services, which is your identity provider. SAP Cloud Identity Services creates an application with the prefix [\*SAP BTP subaccount\*](#) and the display name of your subaccount in the administration console for SAP Cloud Identity Services.

### ❖ Example

If your subaccount was named [\*My Subaccount\*](#), the resulting application in SAP Cloud Identity Services would be [\*SAP BTP subaccount My Subaccount\*](#).

Older applications start with [\*XSUAA\\_\*](#).

### → Tip

To troubleshoot problems with tokens from SAP Cloud Identity Services, see [Logging OpenID Connect Tokens](#) in the documentation for SAP Cloud Identity Services.

### ⚠ Restriction

If the OIDC issuer is changed in the trust configuration, the trust breaks. To adapt the trust configuration on SAP BTP side, use the `btp update security/trust` command of the SAP BTP command line interface together with the `--refresh` parameter. This parameter refreshes the trust configuration to reflect changes in the SAP Cloud Identity Services tenant, for example the issuer value. For more information, see [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#).

## Next Steps

If you don't need the default identity provider anymore, set it to inactive or hide the logon link.

## Related Information

[Default Identity Provider \[page 2258\]](#)

## 6.1.1.8.1.1.1 Map User Attributes from a Corporate Identity Provider for Business Users

When you enable trust with a tenant of SAP Cloud Identity Services, you get an OpenID Connect (OIDC) application in SAP Cloud Identity Services to represent your subaccount, in the context of business users. When SAP Cloud Identity Services authenticates users using a corporate identity provider, map the user attributes provided by the corporate identity provider to the attributes required by your applications.

### Finding the Application for Your Subaccount

The name of the application in the administration console of SAP Cloud Identity Services that represents your subaccount has the prefix SAP\_BTP\_subaccount or XSUAA\_ and the display name of your subaccount.

If your subaccount is named My\_Subaccount, the resulting application in SAP Cloud Identity Services is SAP\_BTP\_subaccount\_My\_Subaccount or XSUAA\_My\_Subaccount.

### Customizing Attribute Mappings

There are several options to customize attribute mappings in SAP Cloud Identity Services, depending on whether identity federation is enabled or disabled.

For more information about identity federation, see [Configure Identity Federation](#).

- When identity federation is disabled, SAP Cloud Identity Services always propagates all the attributes received from the corporate identity provider to all the applications, on a 1:1 basis. You have the following options:
  - Configure the corporate identity provider to directly send the attributes.
  - If needed, use **enriched token claims** or **enriched assertion attributes** (depending on whether the corporate identity provider is connected with SAML or OIDC) to map the attributes sent by the corporate identity provider, to the attribute names needed by SAP BTP.  
**Enriched token claims** or **enriched assertion attributes** add additional attributes, either based on the corporate identity provider attributes (for example, by renaming them) or on static values.
- When identity federation is enabled, SAP Cloud Identity Services doesn't automatically propagate any attributes from the corporate identity provider to the application. This option requires mappings in the SAP Cloud Identity Services application, for each attribute that is needed by the application.  
Use the **Attributes** in the SAP Cloud Identity Services application representing your subaccount.  
SAP BTP expects the following attributes. The default configuration of the trust configuration sets up the values in the following table in the SAP Cloud Identity Services tenant.

Self-Defined Attributes	Identity Directory
email	Email
email_verified	Email Verified
family_name	Last Name

Self-Defined Attributes	Identity Directory
given_name	First Name
groups	Groups
user_uuid	Global user ID

If the corporate identity provider sends user attributes with other names, set the source to the corporate identity provider and the value to the correct attribute name.

### ❖ Example

If your corporate identity provider sends users' last names as the `sn` attribute, add the corporate identity provider as source to the `last_name` attribute with the value `sn`.

Customized Last Name Attribute Configuration in SAP Cloud Identity Services

Attribute Name	Source	Attribute Value
email	Identity Directory	Email
email_verified	Identity Directory	Email Verified
family_name	Identity Directory	Last Name
given_name	Identity Directory	First Name
groups	Identity Directory	Groups
last_name	Corporate Identity Provider	<div style="display: flex; align-items: center;"> <span style="font-size: 1.5em;">❖</span> Example  <span style="margin-left: 10px;">sn</span> </div>
mail	Identity Directory	Email
user_uuid	Identity Directory	Global User ID

For more information, see [User Attributes](#) in the documentation of SAP Cloud Identity Services.

The subject name identifier attribute is used by SAP BTP to uniquely identify the application user.

For more information, see [Configure the Subject Name Identifier Sent to the Application](#) in the documentation of SAP Cloud Identity Services.

## Default Configuration of ID Tokens

In the default configuration, the attributes provided in the ID token issued by SAP Cloud Identity Services are described in the following table.

## Default Attributes of SAP Cloud Identity Services Tokens

User Attribute of SAP Cloud Identity Services	Assertion Attribute	Description
mail	email	E-mail address of the subject. By default, this value is used for the subject name identifier.
		See the table <i>Default Configurations of the Subaccount in SAP Cloud Identity Services</i> following this table.
mailVerified	email_verified	Indicates whether the subject has confirmed their e-mail address. Your identity provider might require users to verify their e-mail address.
lastName	family_name	Last name of the subject.
firstName	given_name	First name of the subject.
companyGroups	groups	Any groups the subject is assigned to in the identity provider.
userUuid	user_uuid	An identifier for a user that's unique across technology layers such as user interface, APIs, and security tokens, as well as across products and lines of businesses contributing to a business process in the Intelligent Enterprise.  Business applications can use this identifier to correlate information about the user. While not necessary for platform users, the attribute doesn't hinder such users either.

### ⓘ Note

SAP Authorization and Trust Management service supports global user identifiers. When SAP Cloud Identity Services sends a global user identifier, it's included in the SAP Authorization and Trust Management service tokens, which means that you can use it in scenarios where you need to use global user identifiers.

For more information, see [Global User ID in Integration Scenarios](#).

## Default Configuration of the SAP Cloud Identity Services Application

In the application of SAP Cloud Identity Services that represents the subaccount, the configuration sets defaults as shown in the following table.

Default Configurations of the Subaccount in SAP Cloud Identity Services

Configuration	Description
Subject name identifier	<p>This attribute is used by the SAP Authorization and Trust Management service to identify the user for authentication.</p> <p>Default value: E-Mail.</p> <p>For more information, see <a href="#">Configure the Subject Name Identifier Sent to the Application</a> in the documentation of SAP Cloud Identity Services.</p>
List of allowed redirect URIs	<p>The list of URIs to which SAP Cloud Identity Services is allowed to redirect from the application that represents your subaccount.</p> <p>Default value: <code>https://&lt;subdomain&gt;.authentication.&lt;landscape&gt;/login/callback/&lt;origin&gt;</code>.</p> <p>For example: <code>https://mysubdomain.authentication.us10.hana.ondemand.com/login.callback/sap.custom</code></p> <p>For more information, see <a href="#">OpenID Connect Application Configurations</a> in the documentation of SAP Cloud Identity Services.</p>
Post logout redirect URIs	<p>The list of URIs to which SAP Cloud Identity Services is allowed to direct users when logging out.</p> <p>Default value: <code>https://&lt;subdomain&gt;.authentication.&lt;landscape&gt;/*</code>.</p> <p>For example: <code>https://mysubdomain.authentication.us10.hana.ondemand.com/*</code></p> <p>For more information, see <a href="#">OpenID Connect Application Configurations</a> in the documentation of SAP Cloud Identity Services.</p>

## Related Information

[Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#)

## 6.1.1.8.1.1.2 Migration from SAML Trust to OpenID Connect Trust with SAP Cloud Identity Services

To use or get the most out of some applications, such as SAP Build Work Zone and SAP Build Apps, your subaccount must use SAP Cloud Identity Services with OpenID Connect (OIDC) as the custom identity provider. This process helps you change an SAML trust configuration to an OIDC configuration with as little impact to your application users as possible, especially in productive subaccounts.

The goal of this process is as follows:

- Preserve user records, including their authorizations, and also all user-related data in applications and services. User IDs don't change. We update the existing trust configuration instead of creating a new one.
- Preserve role collection mappings to groups or other attributes.
- Ensure that applications receive exactly the same information about users as they did before. We provide guidance and verification steps for when you reconfigure authentication along the chain of subaccount, SAP Cloud Identity Services tenant, and potentially corporate identity provider.

### ⚠ Caution

You could achieve a similar result by deleting your current SAML trust configuration and setting up a new trust with OIDC. However, this method can cause an interruption in the service to your business users. Deleting the trust deletes any local shadow users in the subaccount and their role collection assignments. You would need to re-create those users and their role collection assignments. You would also permanently lose any connection to application data for those users. The internal IDs of the re-created users will never be the same as with the original trust configuration.

## Restrictions

In the SAP BTP cockpit under [Custom Identity Provider for Applications](#), there are no trust configurations with protocol OpenID Connect and no more than one with SAML.

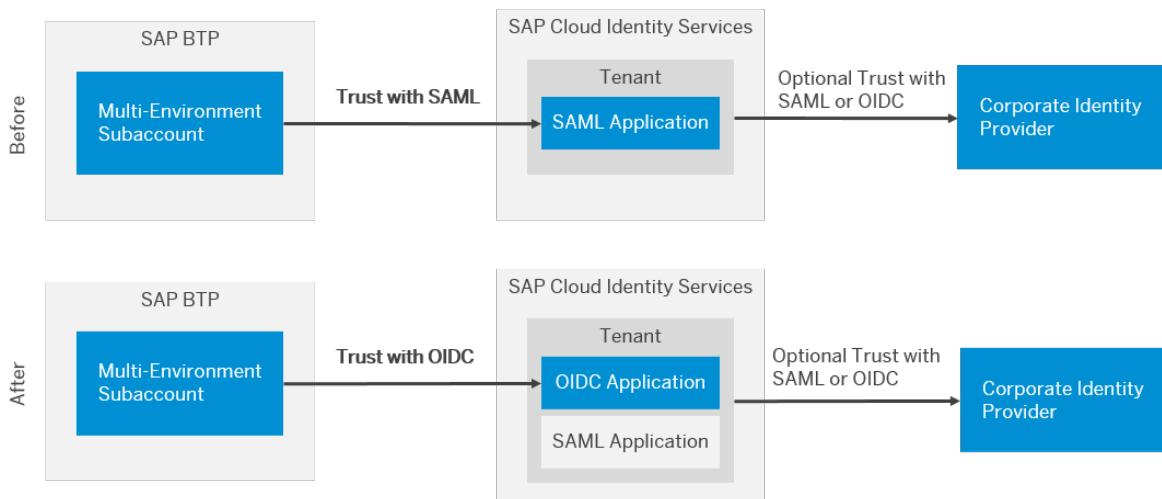
### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

## Architecture

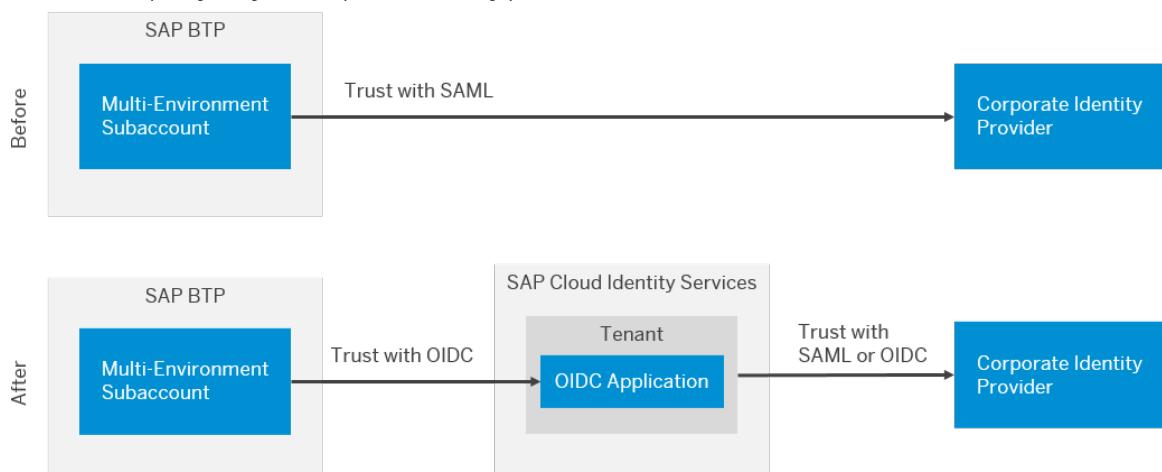
You have one of the following configurations:

- You've already configured SAP Cloud Identity Services as a trusted identity provider, but with SAML. Optionally the SAP Cloud Identity Services tenant trusts a corporate identity provider. For this case, you don't change the architecture, only the protocol between the subaccount and the SAP Cloud Identity Services tenant. If you use a corporate identity provider, configure the new SAP Cloud Identity Services application to authenticate with it.



#### Update of Existing Trust Configuration for SAP Cloud Identity Services

- You've connected your corporate identity provider directly with your subaccount as the trusted identity provider with SAML.  
 For this case, you insert trust to SAP Cloud Identity Services in between and configure SAP Cloud Identity Services as a proxy for your corporate identity provider.



#### Insertion of SAP Cloud Identity Services in Trust Chain of Identity Providers

## Risks and Downtime

Expect a downtime between the time when you start the migration and when you've re-established and configured the chain of trust with your identity providers, including the new SAP Cloud Identity Services application representing your subaccount.

### → Recommendation

Practice this procedure in your dev or test subaccounts before carrying it out in your productive subaccounts. Practice helps you to learn how the migration works, enabling you to minimize the risk of misconfiguration and downtime in your productive environments.

If you use the same SAP Cloud Identity Services tenant for nonproductive and productive subaccounts, the integration between SAP Cloud Identity Services and your corporate identity provider is already complete

as part of the preparation for the nonproductive subaccounts. This setup reduces the risk and downtime for productive subaccounts.

We provide a rollback function if you encounter any problems.

For more information, see [Restore SAML Trust Configuration \[page 2223\]](#).

## Process Overview

1. Prepare for the migration to minimize the downtime starting in step 2.
  1. Collect and save the user data with the existing SAML trust. After the migration, use this information to verify that everything works correctly.
  2. If you use a corporate identity provider and it isn't yet trusted by your SAP Cloud Identity Services tenant, establish trust now.  
For more information, see [Prepare for Migration from SAML Trust to OpenID Connect \[page 2214\]](#).
2. Migrate the subaccount's trust configuration, updating the existing SAML trust to an OIDC trust with SAP Cloud Identity Services.  
For more information, see [Migrate from SAML Trust Configuration to OpenID Connect Trust with SAP Cloud Identity Services \[page 2216\]](#).
3. Configure your new OIDC application for your subaccount.  
Adjust the user subject and any attributes used by your new application in SAP Cloud Identity Services to match your old trust configuration.  
If you use a corporate identity provider, configure the application in SAP Cloud Identity Services to use the corporate identity provider for user authentication.  
For more information, see [Configuration of SAP Cloud Identity Services After Migration from SAML to OIDC \[page 2218\]](#).
4. Verify that user authentication works as before the migration.  
For more information, see [Test the Trust Configuration After the Migration \[page 2221\]](#).

### 6.1.1.8.1.1.2.1 Prepare for Migration from SAML Trust to OpenID Connect

Before migrating your trust configuration, check which user data are sent today, so you can validate the new configuration after the migration. If you use a corporate identity provider, prepare the trust between your SAP Cloud Identity Services tenant and your corporate identity provider ahead of time.

#### Procedure

1. Check what attributes your users have.

To see what attributes your user has, open the following page of your subaccount:

```
https://<subaccount_subdomain>.authentication.<region>.hana.ondemand.com/config?  
action=who&details=true&trustMigrationOnly=true
```

For example

```
https://my-subdomain.authentication.eu10.hana.ondemand.com/config?  
action=who&details=true&trustMigrationOnly=true
```

If at logon you see multiple identity providers, choose the one you want to migrate from for your applications.

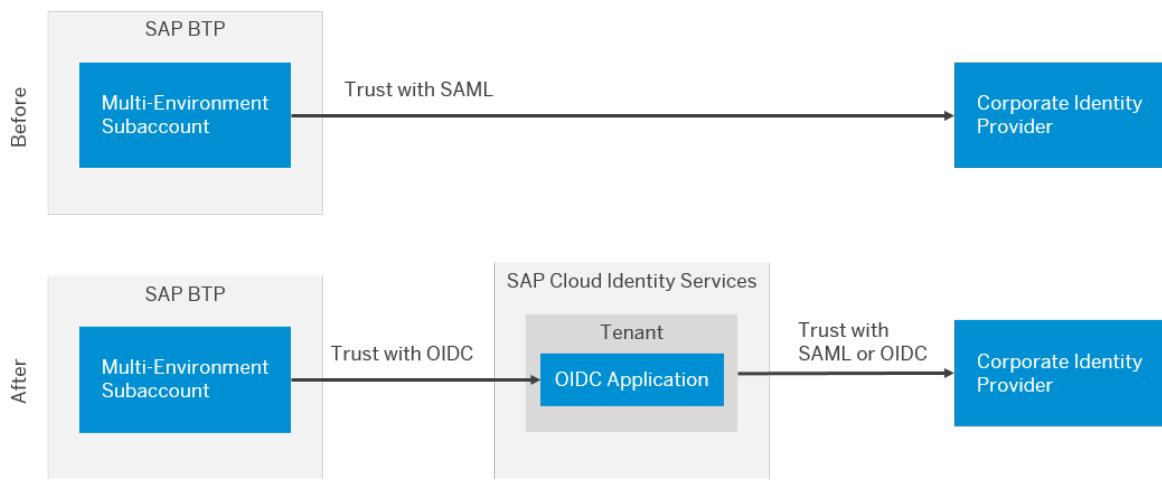
#### → Remember

Save the results. You'll need to compare the attributes with the attributes you get after migrating the trust configuration.

For an example, see [Test the Trust Configuration After the Migration \[page 2221\]](#).

2. Determine if you need to establish trust between your corporate identity provider and your SAP Cloud Identity Services tenant.

If you previously had configured your subaccount trust to your corporate identity provider directly, configure your SAP Cloud Identity Services tenant to trust your corporate identity provider.



In the preceding figure, we're configuring the SAML or OIDC trust between the SAP Cloud Identity Services tenant and the corporate identity provider. Configure the corporate identity provider to send the same value for subject name identifier and the attributes. Keep the attribute names the same, too, except for the attribute names listed in the following table.

From User Attribute	To Assertion Attribute
first_name	given_name
last_name	family_name
mail	email

For more information, see [Corporate Identity Providers](#) in the documentation of SAP Cloud Identity Services.

## 6.1.1.8.1.1.2.2 Migrate from SAML Trust Configuration to OpenID Connect Trust with SAP Cloud Identity Services

If your subaccount uses the SAML protocol to connect to your custom identity provider, you can't consume applications or application features that require direct authentication with SAP Cloud Identity Services, such as SAP Build Work Zone and SAP Build Apps. To consume such applications or features, switch to using OpenID Connect (OIDC) with SAP Cloud Identity Services.

### Prerequisites

- The prerequisites for working with the SAP BTP command line interface (CLI) are fulfilled (see [Log in \[page 2337\]](#)).
- In the SAP BTP cockpit under *Custom Identity Provider for Applications*, there are no trust configurations with the OpenID Connect protocol and no more than one with SAML; namely the configuration that you want to migrate.  
For more information, see [Migration from SAML Trust to OpenID Connect Trust with SAP Cloud Identity Services \[page 2212\]](#).

### Procedure

1. Find the origin key of the identity provider that you want to switch.

You can look up the origin key in the trust configuration in the SAP BTP cockpit using the btp CLI, or the Identity Provider Management API.

Subaccount: testme - Trust Configuration

All: 2

Establish Trust		Manual Setup:	<a href="#">↑ New Trust Configuration</a>	<a href="#">↓ SAML Metadata</a>	Search			
Status	Name	Description	Origin Key	Available for User Logon	SAP BTP Cockpit	SAP BTP CLI	Protocol	Actions
<strong>Default</strong>								
Active	Default identity provider	Default identity provider	sap.default	<input checked="" type="checkbox"/>			OpenID Connect	<a href="#"></a> <a href="#"></a>
<strong>Custom Identity Provider for Applications</strong>								
Active	ar9ibaxhm		my-origin	<input checked="" type="checkbox"/>			SAML	<a href="#"></a> <a href="#"></a>

Finding the Origin Key in the Cockpit

2. To migrate the trust configuration from SAML to OIDC, use the SAP BTP command line interface (CLI).

For more information, see [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#).

3. Log on to the btp CLI (see [Log in \[page 2337\]](#)).
4. Make sure that you set the target to the relevant SAP BTP subaccount.
5. Find the name of the SAP Cloud Identity Services tenant you want to migrate to OpenID Connect. Use the following command:

```
btp list security/available-idp
```

You receive a list of all SAP Cloud Identity Services tenants that are available in this subaccount. It's a good idea to copy the name of the SAP Cloud Identity Services tenant that you want to migrate.

6. Migrate your trust configuration using the following command:

```
btp migrate security/trust ORIGIN --idp TENANT
```

#### ❖ Example

```
btp migrate security/trust my-origin-key --idp myidp.accounts.ondemand.com
```

The btp CLI returns that the trust configuration is active and that the protocol is OIDC.

7. Check that your SAML trust configuration was migrated to a trust configuration with the OpenID Connect protocol. Go to your subaccount in the SAP BTP cockpit and choose .

## Results

In the cockpit, there's a new OIDC trust configuration for your SAP Cloud Identity Services tenant. The original SAML trust configuration has been set to inactive and has received a new `oidc-migration-backup` origin key. If you need to roll back the change, the service uses this configuration to restore your original configuration.

In your SAP Cloud Identity Services tenant, there's a new application that represents the trust configuration. The name is `SAP_BTP_subaccount <subaccount_display_name>`.

## Next Steps

Configure your new SAP Cloud Identity Services application to match your old configuration. Check that your users can log on with all the required attributes. If necessary, configure your new application to consume your corporate identity provider.

#### ⓘ Note

If everything works, you can delete the original SAML trust configuration. If you do so, you can't restore it.

## Related Information

[Configuration of SAP Cloud Identity Services After Migration from SAML to OIDC \[page 2218\]](#)

[Restore SAML Trust Configuration \[page 2223\]](#)

### 6.1.1.8.1.1.2.3 Configuration of SAP Cloud Identity Services After Migration from SAML to OIDC

You replaced an SAML trust configuration to your custom identity provider with an OIDC trust configuration to SAP Cloud Identity Services. Now, you need to make sure that the subaccount gets the same user attributes (names and values) as before.

How you proceed depends on which architecture you're trying to achieve. We provide guidance for the configuration according to your target architecture.

- Business users authenticate directly with SAP Cloud Identity Services  
See [SAP Cloud Identity Services Without Corporate Identity Provider \[page 2218\]](#).
- Business users authenticate with a corporate identity provider. SAP Cloud Identity Services is a proxy:
  - Identity federation is disabled (default flow)
  - Identity federation is enabled (advanced)See [SAP Cloud Identity Services with Corporate Identity Provider \[page 2219\]](#).

#### Related Information

[Migration from SAML Trust to OpenID Connect Trust with SAP Cloud Identity Services \[page 2212\]](#)

[Restore SAML Trust Configuration \[page 2223\]](#)

[Map User Attributes from a Corporate Identity Provider for Business Users \[page 2208\]](#)

### 6.1.1.8.1.1.2.3.1 SAP Cloud Identity Services Without Corporate Identity Provider

You've only changed the protocol of the trust configuration between your SAP Business Technology Platform subaccount and SAP Cloud Identity Services.

Copy the following configurations from the old SAML-based SAP Cloud Identity Services application to your new OIDC-based SAP Cloud Identity Services application:

- Subject name identifier  
See [Configure the Subject Name Identifier Sent to the Application](#) in the documentation of SAP Cloud Identity Services.
- Attributes  
Change the names of the following attributes as listed in the following table.

From User Attribute	To User Attribute
first_name	given_name
last_name	family_name
mail	email

See [Configuring User Attributes from the Identity Directory](#).

## Related Information

[Configuration of SAP Cloud Identity Services After Migration from SAML to OIDC \[page 2218\]](#)

[Test the Trust Configuration After the Migration \[page 2221\]](#)

## 6.1.1.8.1.1.2.3.2 SAP Cloud Identity Services with Corporate Identity Provider

You want to insert your SAP Cloud Identity Services tenant in the user authentication flow between your SAP BTP subaccount and your corporate identity provider. So far, you've migrated the trust configuration from your subaccount to use SAP Cloud Identity Services. Now, you configure SAP Cloud Identity Services to delegate user authentication to the corporate identity provider.

We already asked you to configure trust between your SAP Cloud Identity Services tenant and your corporate identity provider in this step: [Prepare for Migration from SAML Trust to OpenID Connect \[page 2214\]](#).

How you configure the connection depends on whether you're using identity federation or not. Identity federation is a configuration of SAP Cloud Identity Services. If you haven't connected SAP Cloud Identity Services to the corporate identity provider yet, use the disabled approach. If your SAP Cloud Identity Services tenant already trusts your corporate provider, then choose the approach that fits to the existing trust configuration.

For more information about identity federation, see [Configure Identity Federation](#) in the documentation of SAP Cloud Identity Services.

- Identity federation is disabled (default option)  
See [Choose Default Identity Provider for an Application](#) in the documentation of SAP Cloud Identity Services.
- Identity federation is enabled (advanced)  
See [Configure Conditional Authentication for an Application](#) in the documentation of SAP Cloud Identity Services.

## Identity Federation Is Disabled

As you were preparing for this migration, you saw what subject and attributes you need for your users.

For more information, see [Prepare for Migration from SAML Trust to OpenID Connect \[page 2214\]](#).

Attributes are passed to SAP BTP just as they're received from the corporate identity provider. Subject and attribute configurations on the SAP Cloud Identity Services application are ignored. The subject **must** be provided by the corporate identity provider as required. Attributes **ideally** come from the corporate identity provider with the attribute names used by the subaccount so far.

Compare the attributes you got before the migration to the attributes you get now.

For more information, see [Test the Trust Configuration After the Migration \[page 2221\]](#).

If the assertion attributes don't come from the corporate identity provider as you expected, change the names of the following assertion attributes as listed in the following table.

From User Attribute	To User Attribute
first_name	given_name
last_name	family_name
mail	email

Map the attribute names between what the corporate identity provider provides and what the subaccount expects, according to the protocol of the trust configuration:

- For OIDC: See [Enrich Token Claims Coming from Corporate IdP](#) in the documentation of SAP Cloud Identity Services.
- For SAML: [Enrich Assertion Attributes Coming from Corporate IdP](#) in the documentation of SAP Cloud Identity Services.

## Identity Federation Is Enabled

As you were preparing for this migration, you saw what subject and attributes you need for your users.

For more information, see [Prepare for Migration from SAML Trust to OpenID Connect \[page 2214\]](#).

Configure your new OIDC-based SAP Cloud Identity Services application to send the same information to your SAP BTP subaccount as before.

- Subject name identifier  
See [Configure the Subject Name Identifier Sent to the Application](#) in the documentation of SAP Cloud Identity Services.
- Assertion attributes

### → Recommendation

Change the names of the following assertion attributes as listed in the following table.

From User Attribute	To User Attribute
first_name	given_name
last_name	family_name

From User Attribute	To User Attribute
mail	email

See [Configuring User Attributes from a Corporate Identity Provider](#) in the documentation of SAP Cloud Identity Services.

Compare the attributes you got before the migration to the attributes you get now.

For more information, see [Test the Trust Configuration After the Migration \[page 2221\]](#).

## 6.1.1.8.1.1.2.4 Test the Trust Configuration After the Migration

Check that your users can still access their applications and have all the attributes they require. Use these steps to validate your new configuration.

### Prerequisites

You saved the list of attributes you get before the migration.

For more information, see [Prepare for Migration from SAML Trust to OpenID Connect \[page 2214\]](#).

### Procedure

1. Check what attributes your user has now.

```
https://<subaccount_subdomain>.authentication.<region>.hana.ondemand.com/config?
action=who&details=true&trustMigrationOnly=true
```

For example

```
https://my-subdomain.authentication.eu10.hana.ondemand.com/config?
action=who&details=true&trustMigrationOnly=true
```

2. Compare with the results you had before the migration.

In the following example, the user has a few special attributes in addition to the standard ones:

- Custom attribute `cost_center`
- Group assignment `my-group`
- Authorization assignment over the group to the role collection for reading audit logs

Before (SAML)	After (OIDC)
<pre> userName P000011 origin my-origin-key zoneId 9fcfd13d0-77d7-4cfa-86b0- a41b02e662a6 User Attributes {     user_uuid=[7ebf5c32-cacd-4ad2- a208-6bcb3e764f6f],     email=[maria.fontes@example.com],     cost_center=[my-cost-center],     Groups=[my-group],     family_name=[Fontes],     given_name=[Maria] } email maria.fontes@example.com userId 128af760-ad57-4213- b063-2dd381a4d92e externalId P000011 given_name Maria family_name Fontes verified true authorities [     openid,     user_attributes,     auditlog-management! b8.ReadAuditLogs,     uaa.user ] </pre>	<pre> userName P000011 origin my-origin-key zoneId 9fcfd13d0-77d7-4cfa-86b0- a41b02e662a6 User Attributes {     given_name=[Maria],     user_uuid=[7ebf5c32-cacd-4ad2- a208-6bcb3e764f6f],     cost_center=[my-cost-center],     groups=[my-group],     family_name=[Fontes],     email=[maria.fontes@example.com] } email maria.fontes@example.com userId 128af760-ad57-4213- b063-2dd381a4d92e externalId P000011 given_name Maria family_name Fontes verified true authorities [     openid,     user_attributes,     auditlog-management! b8.ReadAuditLogs,     uaa.user ] </pre>

### ⓘ Note

In the new OIDC application, uppercase or lower case G in the `groups` attribute name doesn't matter.

- If there are any remaining differences, adjust the configuration of the SAP Cloud Identity Services tenant or applications or of the corporate identity provider until you get the same attributes.

For more information, see [Configuration of SAP Cloud Identity Services After Migration from SAML to OIDC \[page 2218\]](#).

Use the OIDC token logging in SAP Cloud Identity Services to help troubleshoot any problems.

For more information, see [Logging OpenID Connect Tokens](#) in the documentation of SAP Cloud Identity Services.

If you can't get the configuration to work, restore the SAML trust configuration.

For more information, see [Restore SAML Trust Configuration \[page 2223\]](#).

## Next Steps

Once everything is working as you expected and are certain you don't need it anymore, delete your old SAML trust configuration with the origin `oidc-migration-backup`.

### ⚠ Caution

If you delete the old SAML trust configuration too early, you can't use the restore feature anymore.

## Related Information

[Restore SAML Trust Configuration \[page 2223\]](#)

### 6.1.1.8.1.1.2.5 Restore SAML Trust Configuration

You replaced a SAML trust configuration to your custom identity provider with an OpenID Connect (OIDC) trust configuration to SAP Cloud Identity Services, and the authentication of application users in the subaccount isn't working as you expected. Restore your SAML configuration to get your applications working again.

## Prerequisites

You previously migrated your trust configuration from SAML to OIDC.

For more information, see [Migrate from SAML Trust Configuration to OpenID Connect Trust with SAP Cloud Identity Services \[page 2216\]](#).

## Procedure

- Find the inactive OIDC trust configuration with the `oidc-migration-backup` origin key. There is an active trust configuration with the same name and a different origin key. This is the one you can restore to a SAML trust configuration.

You can look up the origin key in the trust configuration in the SAP BTP cockpit, using the SAP BTP command-line interface (CLI), or the Identity Provider Management API.

Establish Trust								Manual Setup:	<a href="#">↑ New Trust Configuration</a>	<a href="#">↓ SAML Metadata</a>	Search
Status	Name	Description	Origin Key	Available for User Logon	SAP BTP Cockpit	SAP BTP CLI	Protocol	Actions			
<strong>Default</strong>											
Active	Default identity provider	Default identity provider	sap.default	<input checked="" type="checkbox"/>			OpenID Connect	<a href="#"></a>	<a href="#"></a>		
<strong>Custom Identity Provider for Applications</strong>											
Active	ar9ibaxhm		my-origin	<input checked="" type="checkbox"/>			OpenID Connect	<a href="#"></a>	<a href="#"></a>		
Inactive	ar9ibaxhm		oidc-migration-backup	<input checked="" type="checkbox"/>			SAML	<a href="#"></a>	<a href="#"></a>		

#### Finding the Origin Key in the Cockpit

- To restore the trust configuration from OIDC to SAML, use the SAP BTP command line interface (CLI).

For more information, see [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#).

- Log on to the btp CLI (see [Log in \[page 2337\]](#)).
- Make sure that you set the target to the relevant SAP BTP subaccount.

- Find the name of the (currently active) OIDC trust configuration that you want to restore to the SAML protocol.

 Note

There is also an inactive trust configuration with the same name, but it has the `oidc-migration-backup` origin key and the SAML protocol. It still has the data and assignments of the original SAML trust configuration.

- Restore your trust configuration. Use the original key of the active OIDC trust configuration using the following command:

```
btp restore security/trust my-origin-key
```

The btp CLI returns that the restored trust configuration is active and that the protocol is SAML.

- Check whether your trust configuration with the OpenID Connect protocol has been restored to SAML. Go to your subaccount in the SAP BTP cockpit and choose   [Security > Trust Configuration](#). You see only one SAML trust configuration - the restored one. The OIDC trust configuration has disappeared.

## Results

The original SAML trust configuration has been restored, with its original role collection assignments and origin key. The respective OIDC trust configuration no longer exists. In your SAP Cloud Identity Services tenant, the application for the OIDC trust configuration, `SAP_BTP_subaccount <subaccount_display_name>` has been deleted.

### 6.1.1.8.1.1.3 Manually Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services

Use your SAP Cloud Identity Services tenant as an identity provider or a proxy to your own identity provider hosting your business users. Exchange SAML metadata to establish trust with the SAP Cloud Identity Services tenant and then register your subaccount with the tenant. To complete federation, maintain the federation attributes of the user groups.

## Context

 Note

Instead of manually exchanging SAML metadata, we recommend that you use the automatic method based on the OpenID Connect (OIDC) protocol.

For more information, see [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#).

## 6.1.1.8.1.1.3.1 Establish Trust with an SAML 2.0 Identity Provider in a Subaccount

You want to use an SAML 2.0 identity provider, for example, SAP Cloud Identity Services. The identity provider authenticates business users for SAP BTP.

### Prerequisites

- You have subaccount administrator permissions.
- You have downloaded the SAML 2.0 metadata file from the SAP Cloud Identity Services tenant using the URL `https://<sap_cloud_identity_services_tenant>.accounts.ondemand.com/saml2/metadata`.

#### ❖ Example

```
https://my-ias-tenant.accounts.ondemand.com/saml2/metadata
```

For more information, see [Tenant SAML 2.0 Configuration in the SAP Cloud Identity Services](#).

### Context

The following procedure describes how to establish trust with an SAP Cloud Identity Services tenant.

#### ⚠ Restriction

You've already created a trust configuration with a custom identity provider for applications. In this case, you can't add a trust configuration with the same SAP Cloud Identity Services tenant using another protocol.

#### ❖ Example

If there's already a trust configuration with OpenID Connect, you can't create one using the SAML protocol.

#### ⚠ Restriction

Consider the upper limits for trust configurations in the subaccount. See [Limits for the Subaccount \[page 3089\]](#).

#### ⚠ Restriction

We recommend a single trust configuration per subaccount with SAP Cloud Identity Services using the OpenID Connect (OIDC) protocol. If this isn't sufficient, you may use a trust configuration with SAML 2.0.

Nevertheless, we recommend migrating existing SAML 2.0 trusts according to the documentation [Migration from SAML Trust to OpenID Connect Trust with SAP Cloud Identity Services \[page 2212\]](#).

## Procedure

1. In the SAP BTP cockpit, go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose .
2. Choose [New SAML Trust Configuration](#).
3. Enter a name and a description that make it clear to which identity provider that the trust configuration refers.

### Note

Make sure that the users who are supposed to log on to this identity provider understand the name of the trust configuration.

4. Download the relevant metadata (see prerequisites) and save it in an XML file.
5. Choose the [Upload](#) button to insert the SAML 2.0 metadata.
6. To validate the metadata, choose [Parse](#). The system fills the *Subject* and *Issuer* fields with the relevant data from the SAP Cloud Identity Services - your SAML 2.0 identity provider. You see the fields when you choose [Show Details](#).

The name of the new trust configuration now shows the value `<sap_cloud_identity_services_tenant>.accounts.ondemand.com`. It represents the custom identity provider SAP Cloud Identity Services.

This action also fills the fields for the single sign-on URLs and the single logout URLs.

7. Save your changes.
8. To get the SAML metadata of your subaccount, choose [Download SAML Metadata](#). You download an XML file with the SAML metadata of your subaccount. Its name is `saml-<subdomain>-sp.xml`. Use this file to import the SAML metadata into your identity provider.

## Next Steps

Register your SAP BTP subaccount in the SAML 2.0 Identity Provider

### 6.1.1.8.1.1.3.2 Register SAP BTP Subaccount in the SAML 2.0 Identity Provider

An SAML service provider interacts with an SAML 2.0 identity provider to authenticate users signing in by means of a single sign-on (SSO) mechanism. In this scenario, the SAP Authorization and Trust Management service (XSUAA) acts as a service provider representing a single subaccount. To establish trust between an identity provider and a subaccount, you must register your subaccount by providing the SAML metadata to the identity provider. The identity provider is SAP Cloud Identity Services.

## Context

Administrators must configure trust on both sides, in the service provider and in the identity provider. This description covers the side of the identity provider (SAP Cloud Identity Services). The trust configuration on the side of the SAP Cloud Identity Services must contain the following items:

- Metadata for web-based authentication or the relevant configuration information  
The metadata contains the configuration information, including the signing certificate and the required URLs. Create the metadata file of your subaccount using the [Download SAML Metadata](#) button in [Security](#) [Trust Configuration](#) of your subaccount.
- Use e-mail as the unique name ID attribute, and map the user attribute **Groups** to the assertion attribute **Groups** (case-sensitive). This assertion attribute is required for the assignment of roles.

We illustrate the process of configuring trust in the service provider by describing how administrators use the administration console of SAP Cloud Identity Services to register the subaccount.

To establish trust from a tenant of SAP Cloud Identity Services to a subaccount, assign a metadata file and define attribute details. The SAML 2.0 assertion includes these attributes. With the SAP Authorization and Trust Management service as SAML 2.0 service provider, they're used for automatic assignment of SAP Authorization and Trust Management service authorizations based on information maintained in the identity provider.

## Procedure

1. Open the administration console of SAP Cloud Identity Services.

[https://<sap\\_cloud\\_identity\\_services\\_tenant>.accounts.ondemand.com/admin](https://<sap_cloud_identity_services_tenant>.accounts.ondemand.com/admin)

2. Choose [Applications & Resources](#) in the menu or the [Applications](#) tile.
3. To add a new SAML service provider, choose [Create](#).
4. Enter the required data and choose [+ Create](#).
  - Enter a name for the application that clearly identifies it as your new service provider.  
Users see this name in the logon screen when the authentication is requested by the SAP Authorization and Trust Management service. Seeing the name, they know which application they currently access after authentication.
  - Choose [SAML 2.0](#) as the protocol type.
5. Choose [Trust](#) [SAML 2.0 Configuration](#).
6. Under [Define from Metadata](#), upload the relevant metadata XML file and choose [Save](#).

### Note

Use the `saml-<subdomain>-sp.xml` metadata file of your subaccount. You can create the file using the [Download SAML Metadata](#) button in [Security](#) [Trust Configuration](#) of your subaccount.

If the contents of the metadata XML file are valid, the parsing process extracts the information required to populate the remaining fields of the SAML configuration. It provides the name, the URLs of the assertion consumer service and single logout endpoints, and the signing certificate.

7. Choose *Default Name ID Format* and select *E-Mail* as a unique attribute. Save your changes.
8. Choose *Attributes*, use *Add* to add an attribute named **Groups** with a source **Identity Provider** and the value **Groups** (case-sensitive). Save your changes.

 Note

Be aware that the X.509 certificates exposed in the SAML 2.0 metadata have specific validities. This applies to both SAML service providers and SAML identity providers. Authentication may fail with an error once the X.509 certificates expire.

Ensure that the SAML 2.0 metadata is updated before the X.509 certificate expiry dates.

Sign the certificate with a new signing key. For more information, see [Rotate Signing Keys of SAML Token \[page 2312\]](#).

#### 6.1.1.8.1.1.4 Establish Trust and Federation with UAA Using Any SAML Identity Provider

Integrate trust between the SAP BTP subaccount and any SAML 2.0 identity provider.

#### Context

 Tip

Avoid multiple trust configurations in the same subaccount. We recommend that you use SAP Cloud Identity Services as identity provider and connect a potential corporate identity provider there. We strongly recommend OpenID Connect (OIDC). If you use SAML 2.0, connect an SAP Cloud Identity Services tenant and then add your corporate identity provider. If you don't have a tenant yet, check [Getting a Tenant](#).

For more information, see the announcement of December 31, 2024 in the [release notes](#).

To establish trust, configure the trust configuration of the SAML 2.0 identity provider in your subaccount using the SAP BTP cockpit. Next, register your subaccount in User Account and Authentication service using the administration console of your SAML 2.0 identity provider. To complete federation, maintain the federation attributes of the SAML 2.0 user groups. This makes sure that you can assign authorizations to user groups.

## 6.1.1.8.1.1.4.1 Establish Trust with Any SAML 2.0 Identity Provider in a Subaccount

You want to use an SAML 2.0 identity provider. This is where the business users for SAP BTP are stored.

### Prerequisites

- You have subaccount administrator permissions.
- You have downloaded the SAML 2.0 metadata file from your SAML 2.0 identity provider using the relevant URL. For more information, see the documentation of your identity provider.

### Context

You want to establish a trust relationship with an SAML 2.0 identity provider in your subaccount in SAP BTP. The following procedure guides you through the trust configuration.

### Procedure

1. Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose  [Security](#)  [Trust Configuration](#) in the SAP BTP cockpit.
2. Choose [New Trust Configuration](#).
3. Enter a name and a description that make it clear that the trust configuration refers to your identity provider.

#### Note

Make sure that the users who are supposed to log on to this identity provider understand the name of the trust configuration.

4. To get the relevant metadata, get to the metadata of your identity provider.
5. Copy the SAML 2.0 metadata and paste it into the [Metadata](#) field.
6. To validate the metadata, choose [Parse](#). Take care that the [Subject](#) and [Issuer](#) fields are filled with the relevant data from your SAML 2.0 identity provider.

The name of the new trust configuration now shows the value of your identity provider. It represents your custom identity provider.

Check whether the fields for the single sign-on URLs and the single logout URLs are filled.

7. Save your changes.
8. To get the SAML metadata of your subaccount, choose the [SAML Metadata](#) button.

You download an XML file with the SAML metadata of your subaccount. Its name is `saml-<subdomain>-sp.xml`. Use this file to import the SAML metadata into your identity provider.

## 6.1.1.8.1.1.4.2 Register SAP BTP Subaccount in Any SAML 2.0 Identity Provider

An SAML service provider interacts with an SAML 2.0 identity provider to authenticate users signing in by means of a single sign-on (SSO) mechanism. In this scenario, the User Account and Authentication (UAA) service acts as a service provider representing a single subaccount. To establish trust between an identity provider and a subaccount, you must register your subaccount by providing the SAML details for web-based authentication in the identity provider itself.

### Context

Administrators must configure trust on both sides, in the service provider and in the SAML identity provider. This description tries to guide you through the configuration of your identity provider. The trust configuration on the side of the SAM 2.0 identity provider must contain the following items:

- Metadata for web-based authentication or the relevant configuration information  
If available, the metadata contains the configuration information, including the signing certificate and the required URLs. You can create the metadata file of your subaccount using the [SAML Metadata](#) button in [Security](#) [Trust Configuration](#) of your subaccount.
- Use e-mail as the unique name ID attribute, and map the user attribute **Groups** to the assertion attribute **Groups** (capitalized). This attribute is required for the assignment of roles.  
This makes sure that there is a trust relationship between your SAM 2.0 identity provider and the subaccount.

Your administrators use the administration console of your SAML 2.0 identity provider to register the subaccount.

To establish trust from a tenant of your identity provider to a subaccount, assign a metadata file and define attribute details. The SAML 2.0 assertion includes these attributes. With the UAA as SAML 2.0 service provider, they are used for automatic assignment of UAA authorizations based on information maintained in the identity provider.

### Procedure

1. Open the administration console of your SAML 2.0 identity provider.
2. Go to the configuration of your SAML 2.0 identity provider.
3. Add a new SAML service provider as described in the documentation of your identity provider.
4. (If applicable) Choose a name for your identity provider that clearly identifies it as your new service provider. Save your changes.

Users see this name in the logon screen when the authentication is requested by the UAA service. Seeing the name, they know which application they currently access after authentication.

5. Choose the SAML 2.0 configuration and import the relevant metadata XML file. Save your changes.

**ⓘ Note**

Use the `saml-<subdomain>-sp.xml` metadata file of your subaccount. You can create the file using the **SAML Metadata** button in **Security** **Trust Configuration** of your subaccount.

Take care that the fields of the SAML configuration are filled. The metadata provides information, such as the name, the URLs of the assertion consumer service and single logout endpoints, and the signing certificate.

6. Choose or create the name ID attribute and select E-mail as a unique attribute. Save your changes.
7. Choose or create a user attribute, and enter **Groups** (capitalized) as attribute name for the **Groups** user attribute. Save your changes.

## Related Information

[Settings for Default SAML Federation Attributes of Identity Providers for Business Users \[page 2231\]](#)

### 6.1.1.8.1.1.4.3 Settings for Default SAML Federation Attributes of Identity Providers for Business Users

This table shows the attribute settings of the identity provider and the values administrators use to establish trust between the SAML 2.0 identity provider and a subaccount.

Since there are multiple identity providers you can use, we display the parameters and values of SAP Cloud Identity Services. Use the information in this table as a reference for the configuration of your identity provider.

[Settings of SAP Cloud Identity Services for SAML 2.0 Trust](#)

Section in the User Interface	Description	Recommended Value
<a href="#">SAML 2.0 Configuration</a>	Configures the trust with a service provider using an uploaded metadata file.	Upload the metadata file from the SAP Authorization and Trust Management service.

Section in the User Interface	Description	Recommended Value
<i>Default Name ID Format</i>	<p>Configures the attribute that the identity provider uses to identify the users. The attribute is sent as the name ID for the authenticated user in SAML assertions.</p> <p>Possible settings:</p> <ul style="list-style-type: none"> <li>• <i>User ID</i></li> <li>• <i>E-Mail</i></li> <li>• <i>Display Name</i></li> <li>• <i>Login Name</i></li> <li>• <i>Employee Number</i></li> </ul>	<p><i>E-Mail</i></p> <p><b> ⓘ Note</b></p> <p>We recommend that you use exactly this name ID format.</p>
<i>Attributes</i>	<p>To assign authorizations indirectly, provide the user groups in the assertion attribute <b>Groups</b> (capitalized).</p> <p>You can change the default names of the assertion attributes that the application uses to recognize the user attributes. You can use multiple assertion attributes for the same user attribute.</p> <p>Some possible settings:</p> <ul style="list-style-type: none"> <li>• <i>Groups</i></li> <li>• <i>first_name</i></li> <li>• <i>last_name</i></li> <li>• <i>email</i></li> </ul> <p>You can choose from a number of user attributes and add them.</p>	<p>Select the <b>Groups</b> user attribute and enter <b>Groups</b> as assertion attribute. You must set this attribute to enable that the assignment of role collection to user groups. For more information, see the related link.</p> <p><b>⚠ Caution</b></p> <p>Use exactly this spelling: <b>Groups</b></p>

## ⌚ Example

In the following, you see what John Doe's SAML 2.0 assertion looks like if *Default Name ID Format* was set to *E-Mail* and *Assertion Attribute* to *Groups*.

## ↔ Sample Code

```
<Assertion xmlns="urn:oasis:names:tc:SAML:2.0:assertion"
  xmlns:ns2="http://www.w3.org/2000/09/xmldsig#" xmlns:ns3="http://www.w3.org/
  2001/04/xmlenc#" ID="A-de4246c0-397e-4df0-baf0-cde399d55422"
  IssueInstant="2017-10-23T08:52:39.494Z" Version="2.0">
  <Issuer>company-security.accounts.ondemand.com</Issuer>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    ...
    </ds:Signature>
    <Subject>
      <NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress">john.doe@example.com</NameID>
      <SubjectConfirmation
        Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
```

```

        <SubjectConfirmationData
InResponseTo="a3h3d679ae07c8cj2ih97d22i7d79a0"
NotOnOrAfter="2017-10-23T09:02:39.494Z"
Recipient="https://authentication.example.hana.ondemand.com/saml/SSO/alias/
company-prod-example"/>
    </SubjectConfirmation>
</Subject>
<Conditions NotBefore="2017-10-23T08:47:39.494Z"
NotOnOrAfter="2017-10-23T09:02:39.494Z">
    <AudienceRestriction>
        <Audience>aws-live-eu10</Audience>
    </AudienceRestriction>
</Conditions>
<AuthnStatement AuthnInstant="2017-10-23T08:52:39.494Z"
SessionIndex="S-SP-83d68a13-25fe-44b8-a139-649ca3e99363"
SessionNotOnOrAfter="2017-10-23T20:52:39.494Z">
    ..
</AuthnStatement>
<AttributeStatement>
    <Attribute Name="Groups">
        <AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">iot-acme-monitoring</AttributeValue>
        <AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">new-group-01</AttributeValue>
        <AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">new-group-02</AttributeValue>
    </Attribute>
    <Attribute Name="email">
        <AttributeValue xmlns:xs="http://www.w3.org/2001/
XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="xs:string">john.doe@example.com</AttributeValue>
    </Attribute>
</AttributeStatement>
</Assertion>

```

## Related Information

[Map Role Collections to User Groups \[page 2281\]](#)

### 6.1.1.8.1.1.5 Using Multiple Identity Providers from the Same Subaccount

You need to use multiple identity providers for different groups of business users. You want to guide business users to the right identity provider for logon.

Usually there's a single identity provider, which is needed for most users.

Additional identity providers are only needed in exceptional cases.

## Basic Considerations Before You Provide a Logon Link for Business Users

Recommendation	Description
(Preferred)  Try to avoid multiple trust configurations at all.	<p>Hide the default identity provider and connect a single SAP Cloud Identity Services tenant.</p> <p>Connect further identity providers as corporate identity providers in the SAP Cloud Identity Services tenant and use conditional authentication rules in SAP Cloud Identity Services.</p>
(Feasible)  If multiple trusts are really needed in the subaccount, avoid that business users must pick a specific trust by giving them dedicated application URLs, which choose the right identity provider (only supported by some applications).	<p>Some applications can be accessed using a URL that includes a query parameter to define the trust configuration for user login, instead of asking the user to choose one.</p> <p><b>Example</b></p> <pre>https:// application.cfapps.eu10.hana.ondemand .com/sap_idp=some-origin-key</pre>
	<p>Options</p> <ol style="list-style-type: none"><li>1. Make sure all users access applications with such URLs.</li><li>2. Choose one standard trust configuration as the one to be used when applications are accessed without such a query parameter. Only use the parameter when any other trust configuration should be used. To achieve this, disable <a href="#">Available for User Logon</a> for all trust configurations other than the one you want to use as standard trust configuration.</li></ol> <p>For more information, see <a href="#">Provide Logon Link to Identity Provider for Business Users [page 2235]</a>.</p> <p>If your custom applications use application router, you can enable support for the parameter by following the application router documentation. See <a href="#">Dynamic Identity Provider Configuration</a> or <a href="#">routes [page 425]</a>.</p>
(Exceptional)  If business users must really choose the identity provider. Make sure the respective link texts on the login page makes sense to them, so that they can easily choose the right one.	<p>Review the link texts of all trust configurations, where <a href="#">Available for User Logon</a> is enabled, so that business users understand which one to choose.</p> <p>For more information, see <a href="#">Rename the Logon Link Text for Custom Identity Providers [page 2237]</a>.</p>

## 6.1.1.8.1.1.5.1 Provide Logon Link to Identity Provider for Business Users

You want to provide an understandable link on the logon page and guide business to the right identity provider.

You want to make life easy for business users and provide a logon link that they can easily recognize as such. This link provides an identity provider logon that enables them to log on to the application that they want to work with. To make it even easier, you can hide the logon of the default trust configuration (`SAP ID service`) for platform users, although it remains active. That way, your business users know at once which link to use for their logon.

We strongly recommend to use the following procedure:

1. Hide the default identity provider (`SAP ID service`) from the logon page.  
[Hide Logon Link for Default Identity Provider \[page 2235\]](#)
2. Display the custom identity provider at logon time.  
[Display Logon Link for Custom Identity Provider for Business Users \[page 2236\]](#)
3. (Optional) Give the logon link of the custom identity provider a name the business users understand.  
[Rename the Logon Link Text for Custom Identity Providers \[page 2237\]](#)

## 6.1.1.8.1.1.5.2 Hide Logon Link for Default Identity Provider

You use one or multiple custom identity providers for business users as well as the default identity provider primarily for platform users. To provide a good logon experience for your business users, you want to hide the default identity provider, which remains active.

### Prerequisites

- You have configured a custom trust configuration for a custom identity provider, for example SAP Cloud Identity Services, and set it to active.
- You've checked the *Available for User Logon* checkbox in your custom trust configuration.
- The default trust configuration (`SAP ID service`) is active.

## Context

To hide the default identity provider at logon time, proceed as follows:

## Procedure

1. Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose  [Security](#)  [Trust Configuration](#) in the SAP BTP cockpit.  
You see the list of trust configurations: the default trust configuration and the custom trust configuration(s).
2. Choose  (Edit) for the default trust configuration. It has the status *Default*.
3. To hide the default trust configuration (SAP ID service) for logon, uncheck the *Available for User Logon* checkbox.
4. Save your changes.

The default trust configuration is now hidden but remains active.

### 6.1.1.8.1.1.5.3 Display Logon Link for Custom Identity Provider for Business Users

You want to display a logon link of the custom identity provider that business users should use to log on to an application.

## Context

## Procedure

1. Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose  [Security](#)  [Trust Configuration](#) in the SAP BTP cockpit.  
You see the list of trust configurations: the default trust configuration and the custom trust configuration(s).
2. Go to the desired trust configuration with the status *Custom*.
3. Choose  (Edit) for the custom trust configuration.
4. Make sure that the status of your custom trust configuration is *Active*.

5. Check the *Available for User Logon* checkbox.
6. Save your changes.

## Results

Whenever business users want to log on to their application, they see a logon link for the custom identity provider that they should use.

### 6.1.1.8.1.1.5.4 Rename the Logon Link Text for Custom Identity Providers

You can provide a logon link and give the link a name the business users understand. That way, they know which link they should use to log on.

#### Prerequisites

- You have already made your custom identity provider available for user logon (see the related link).

#### Context

You want to define an understandable name for the logon link of the custom identity provider that the business users should use.

#### Procedure

1. Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose   **Security**  **Trust Configuration** in the SAP BTP cockpit.

You see the list of trust configurations: the default trust configuration and the custom trust configuration(s).
2. Choose the link of your custom identity provider. It has the *Custom* status.
3. Choose *Edit*.
4. Go to *Link Text for User Logon* and give the logon link an understandable name.
5. Save your changes.

## Results

Whenever business users want to log on to their application, they see the logon link of the custom identity provider that they should use. It has the caption you entered in [Link Text for User Logon](#).

## Related Information

[Display Logon Link for Custom Identity Provider for Business Users \[page 2236\]](#)

### 6.1.1.8.1.2 Platform Users

**User accounts** enable users to log on to SAP BTP, access subaccounts, and to use applications according to the permissions granted to them. In this context, it's important to understand the difference between the two types of users that we refer to: Platform users and business users.

**Platform users** are usually developers, administrators or operators who deploy, administer, and troubleshoot accounts, applications and services on SAP BTP.

Platform users who were added as members and have administrative permissions can view or manage the list of global accounts, directories, subaccounts, and Cloud Foundry orgs and spaces that are available to them. Members access them using the SAP BTP Cockpit, the SAP BTP command-line interface (btp CLI), or the Cloud Foundry command-line interface (CF CLI).

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

For platform users, there's a [default identity provider \[page 2258\]](#). We expect that you have your own user base. We recommend that you configure SAP Cloud Identity Services as the custom identity provider and connect SAP Cloud Identity Services to your own corporate identity provider.

## Related Information

[Business Users \[page 2205\]](#)

[Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#)

[Supported Tools and Services When Using Custom Identity Providers for Platform Users \[page 2248\]](#)

[Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface \[page 2251\]](#)

[Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#)

## 6.1.1.8.1.2.1 Establish Trust and Federation of Custom Identity Providers for Platform Users

You want to use a custom identity provider for the platform users of SAP BTP in different environments and at the different account levels: global account, directory, and subaccount. By default, platform users in multi-environment subaccounts are users in the default identity provider.

### Prerequisites

- You've a tenant of SAP Cloud Identity Services.  
For more information, see [Tenant Model and Licensing](#) in the documentation for SAP Cloud Identity Services.
- The SAP Cloud Identity Services tenant is associated with the customer IDs of the relevant global account of SAP BTP.  
For more information, see [Reuse SAP Cloud Identity Services Tenants for Different Customer IDs](#) in the documentation for SAP Cloud Identity Services.
- Make sure that the email addresses of all users in your identity provider are unique and correct.

#### ⓘ Note

The email address of the user is the user identifier for all account levels (global account, directory, multi-environment subaccount) and for the Cloud Foundry environment.

### Context

#### ⓘ Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

Platform users perform technical development, deployment, and administration tasks. For example, they perform subaccount administration in the SAP BTP cockpit or access the Cloud Foundry command line interface (CF CLI). By hosting these users in your own identity provider, you gain a number of advantages over hosting them in the default identity provider.

- Integrate the management of these users with your corporate identity management strategy, hosted on your own identity providers. You control your own user lifecycle and single sign-on strategies throughout the entire landscape.
- Enforce your own password and authentication policies, such as stronger passwords or multifactor authentication.

#### ⓘ Note

The content in this section is only relevant for **platform users** and **not** business users.

For more information about establishing trust for business users, see [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#).

You must establish a trust relationship with a custom identity provider in your global account in SAP BTP. The following procedure guides you through the trust configuration in your custom identity provider.

## Procedure

1. Go to your global account (see [Navigate in the Cockpit \[page 2142\]](#)) and choose  [Security](#)  [Trust Configuration](#) in the SAP BTP cockpit.
  2. Choose [Establish Trust](#).
- The identity providers listed are the SAP Cloud Identity Services tenants associated with your customer ID.
3. Select an identity provider from the list of available tenants and choose [Next](#).
  4. Choose your desired domain of the tenant and continue with [Next](#). For a good single sign-on experience, choose the same domain for all SAP BTP accounts and non SAP BTP applications that use this tenant.
  5. Enter a name and a description of the new trust configuration. If possible, set the origin key. The origin key always ends with `-platform`. Continue with [Next](#).

### Note

The origin key is a technical identifier of an identity provider for platform users. Administrators need it when managing users. Platform users from the identity provider need the origin key when signing in with certain tools, such as the Cloud Foundry command line interface or service dashboards.

You can only set the origin key once for the whole landscape. You can't change the origin key later.

6. The wizard shows you a preview of your configuration. To complete your new trust configuration, choose [Finish](#).

### Remember

If the OIDC issuer was changed in the trust configuration, the trust breaks, but the global account automatically repairs the trust after 24 hours.

To adapt the trust configuration on SAP BTP side, use the `btp update security/trust` command of the SAP BTP command line interface together with the `--refresh` parameter. This parameter immediately refreshes the trust configuration to reflect changes in the SAP Cloud Identity Services tenant, for example the issuer value. For more information, see [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#).

## Results

You've configured trust in your tenant of SAP Cloud Identity Services, which is your identity provider. SAP Cloud Identity Services creates an application with the name [SAP Business Technology Platform](#).

### Tip

To troubleshoot problems with tokens from SAP Cloud Identity Services, see [Logging OpenID Connect Tokens](#) in the documentation for SAP Cloud Identity Services.

## Next Steps

- Add platform users to your orgs and spaces.  
For more information about adding members with the SAP BTP cockpit, see [Add Org Members \[page 2437\]](#) and [Add Space Members \[page 2444\]](#).  
For more information about adding members with the command-line interface, see [Add Organization Members Using the Cloud Foundry Command Line Interface \[page 2483\]](#) and [Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#).
- Add platform users at different levels (global account, directory, subaccount) from the custom identity provider.  
For more information, see [Working with Users \[page 2265\]](#).

### ⓘ Note

For subaccounts in the Neo environment, the identity provider will be offered in the value help only if a user in that identity provider has created at least one Neo subaccount in the corresponding global account and Neo region.

### → Tip

If the identity provider isn't available in the value help for Neo subaccount members, log on to the global account with a user from that identity provider and create a new Neo subaccount. If it's not needed otherwise, you can delete it.

### → Recommendation

We recommend that you keep at least one global account administrator from the default identity provider. You can then use this administrator to log on in the rare instance that access to the custom identity provider fails.

- Log on to the SAP BTP cockpit as a user from the custom identity provider. In the *Trust Configuration* page, the *Open* link in the *SAP BTP Cockpit* column contains the URL (for example, <https://emea.cockpit.btp.cloud.sap/cockpit/?idp=cidppuxhm.accounts.ondemand.com>) for the user to log on with the custom identity provider. Copy the link and send it to your platform user colleagues. Remember if you have the cockpit open and if you want to work in parallel, your current session may be shared by your browser. Open the link in a private or incognito browsing mode.  
For more information, see [Log On with a Custom Identity Provider to the SAP BTP Cockpit \[page 2255\]](#).
- If you want to impose, for example, two-factor authentication for platform users, you must configure [two-factor authentication](#) in **all** the SAP Cloud Identity Services applications involved. Multifactor authentication is just used as an example.  
For more information, see [Multi-Factor Authentication](#) in the documentation for the SAP Cloud Identity Services service. Keep all settings for SAP Cloud Identity Services applications the same.

### → Tip

Instead of configuring two-factor authentication in all the SAP Cloud Identity Services applications, you could also configure it once in your corporate identity provider.

### Note

For all the Neo datacenters where you have subaccounts, you might have to configure a separate SAML application (for example, [SAP Cloud Platform hhw9g3jrx](#)) in SAP Cloud Identity Services.

- Integrate your SAP Cloud Identity Services tenant with your identity authentication management solution. For more information, see [Corporate Identity Provider](#) in the documentation for SAP Cloud Identity Services.
- Work in the SAP BTP command line interface (btp CLI).  
To log on to a global account, you need to provide the hostname of the tenant (for example, `ar9ibaxhm`) with the `--idp` parameter. See [Log in with a Custom Identity Provider \[page 2342\]](#).  
To work with users and role collections on global account, directory, or subaccount level, you need to provide the origin key with the `--of-idp` parameter in the following commands: `btp list security/user`, `btp get security/user`, `btp delete security/user`, `btp assign security/role-collection`, `btp unassign security/role-collection`.  
Both values can be found in the cockpit under [Security → Trust Configuration → Custom Platform Identity Providers](#).  
As an administrator, you want to determine which of the SAP Cloud Identity Services tenants' domains SAP BTP should use for platform user logon. For this reason, you specify a custom domain for an SAP Cloud Identity Services tenant. You can use the `btp update security/trust` command and specify the domain in the `--domain` parameter.
- Work in the Cloud Foundry command line interface (CF CLI).  
For more information, see [Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface \[page 2251\]](#).
- The Kyma environment has a separate configuration for the identity provider. We recommend that you also configure your Kyma environment to use SAP Cloud Identity Services as your custom identity provider.  
For more information, see [Configure a Custom Identity Provider for Kyma](#).

## Related Information

[User and Member Management \[page 104\]](#)

## 6.1.1.8.1.2.1.1 Restrictions When Using Custom Identity Providers for Platform Users

The following is a list of restrictions that apply to the use of custom identity providers with platform users.

### Supported with Restrictions When Using Custom Identity Providers for Platform Users

Supported with Restrictions	Description
Maximum number of parallel sessions per user per identity provider	Each user is allowed a maximum of 10 parallel sessions, per identity provider. This number considers all tools, including the cockpit and CLIs.  ⓘ Note  When accessing the cockpit, a user is allowed one session in each region. For example, if you access <a href="https://emea.cockpit.btp.cloud.sap">https://emea.cockpit.btp.cloud.sap</a> , it counts as the first session and <a href="https://amer.cockpit.btp.cloud.sap">https://amer.cockpit.btp.cloud.sap</a> as the second one.
Single logout (SLO)	For platform users of custom identity providers, logging out from the SAP BTP cockpit (including Neo cockpit) terminates the session in the used SAP Cloud Identity Services tenant and sessions of other applications that connect to the same tenant. What is required is that the sessions support this kind of logout. This requirement doesn't apply for other instances of the SAP BTP cockpit except for instances where the user initially logged out from. In this case, sessions remain active.

### Supported with Restrictions for Neo Subaccounts Before the Changes in the Trust Configuration

The following is a list of restrictions that only apply for Neo subaccounts when using custom identity providers for platform users.

- All individual Neo subaccounts that have been created before July 2023.
- Neo subaccounts in global accounts that have custom identity providers for platform users. For these subaccounts, [SAP Note 3330671](#) hasn't been applied yet.

Supported with Restrictions	Description
Working with custom domains for an SAP Cloud Identity Services tenant	<p>SAP BTP always uses the default domain of the SAP Cloud Identity Services tenant, regardless of a potentially configured custom domain. Therefore, when you use this tenant as a platform identity provider:</p> <ul style="list-style-type: none"> <li>Single sign-on (SSO) doesn't work between applications that use this custom domain and cloud management tools. Exception: if you use the same SAP Cloud Identity Services tenant for both platform and business users, as custom domain is a tenant setting.</li> <li>The OpenID Connect (OIDC) issuer in the <a href="#">Name</a> field of the SAP Cloud Identity Services tenant must be the default domain (&lt;origin&gt;.accounts.ondemand.com).</li> </ul> <p>For more information, see <a href="#">Tenant OpenID Connect Configurations</a>.</p>
OpenID Connect (OIDC) issuer in the <a href="#">Name</a> field of the SAP Cloud Identity Services tenant	<p><b>⚠ Caution</b></p> <p>Don't change the <a href="#">Name</a> field after configuring trust. Changing the issuer breaks the trust between the systems.</p>
Neo CLI	<p>No restriction for new Neo subaccounts if this <a href="#">prerequisite</a> is fulfilled.</p> <p>For basic authentication, the Neo CLI has limited support for existing Neo subaccounts when using custom identity providers for platform users.</p>
Neo Git service	Logging on with a password to the Neo Git service doesn't work with custom identity providers for platform users.
Cloud connector	Logging on with a password to the Cloud connector doesn't work with custom identity providers for platform users.
SAP HANA studio	No restriction for new Neo subaccounts if this <a href="#">prerequisite</a> is fulfilled.

## 6.1.1.8.1.2.1.2 Map User Attributes from a Corporate Identity Provider for Platform Users

When you enable trust with a tenant of SAP Cloud Identity Services, you get an OpenID Connect (OIDC) application in SAP Cloud Identity Services to represent SAP BTP, in the context of platform users. When you authenticate users using a corporate identity provider, map the user attributes provided by the corporate identity provider to the attributes required by SAP BTP. The following information explains which attributes SAP BTP needs for which purpose, and how you can map those attributes.

## Prerequisites

- Ensure that your corporate identity provider allows local users from SAP Cloud Identity Services. If local users are not allowed, you must enrich the attributes coming from the corporate identity provider. For more information, see [Enrich Assertion Attributes Coming from Corporate IdP](#).

## Finding the Application for Your Platform Users

The name of the application in the administration console of SAP Cloud Identity Services that represents SAP BTP in the context of platform users has the name, *SAP Business Technology Platform*.

For more information, see [OpenID Connect](#) in the documentation of SAP Cloud Identity Services.

## Customizing Attribute Mappings

There are several options to customize attribute mappings in SAP Cloud Identity Services, depending on whether identity federation is enabled or disabled. For more information about identity federation, see [Configure Identity Federation](#).

- When identity federation is disabled, SAP Cloud Identity Services always propagates all the attributes received from the corporate identity provider to all the applications, on a 1:1 basis. You have the following options:
  - Configure the corporate identity provider to directly send the attributes (see the table below).
  - If needed, use **enriched token claims** or **enriched assertion attributes** (depending on whether the corporate identity provider is connected with SAML or OIDC) to map the attributes sent by the corporate identity provider, to the attribute names needed by SAP BTP.  
**Enriched token claims** or **enriched assertion attributes** add additional attributes, either based on the corporate identity provider attributes (for example, by renaming them) or on static values.
- When identity federation is enabled, SAP Cloud Identity Services doesn't automatically propagate any attributes from the corporate identity provider to the application. This option requires mappings in the SAP Cloud Identity Services application, for each attribute that is needed by the application.  
Use the attributes in the SAP Cloud Identity Services application representing SAP BTP.

### Note

You can add attribute sources or disable the default application attributes. You can also add self-defined attributes for mapping to role collections.

For example, you have so many groups being added to your token and you're running into size limits. You can disable the standard groups configuration and add a regular expression to include only those groups, which begin with BTP. So, to the *groups* attribute, you add a source of type **Expression** with the value  `${companyGroups:regex[BTP.*]}` .

To check which groups SAP Cloud Identity Services actually sends, use the troubleshooting logs for OpenID Connect. For more information, see [Logging OpenID Connect Tokens](#) in the documentation for SAP Cloud Identity Services.

### Note

Ensure that you enter the accurate value names for the attributes as they are provided by your corporate identity provider.

Default Configuration of Application Attributes in SAP Cloud Identity Services

Attribute Name	Source	Attribute Value
email	Identity Directory	Email
	Corporate Identity Provider	email
groups	Identity Directory	Groups
mail	Identity Directory	Email
	Corporate Identity Provider	email
uuid	Identity Provider	User ID

If the corporate identity provider sends user attributes for email address, first and last name with other names than `mail`, `first_name`, or `last_name`, set the right attribute name by replacing those values.

### Example

If your corporate identity provider sends users' last names as the `sn` attribute, add the corporate identity provider as source to the `last_name` attribute with the value `sn`.

For more information, see [User Attributes](#) in the documentation of SAP Cloud Identity Services.

The following table provides the information needed for mapping the attributes.

The subject name identifier attribute is used by SAP BTP to uniquely identify the user in Neo subaccounts. In the Cloud Foundry environment, the email address is used as the user identifier for the global account, directory, and multi-environment subaccount.

For more information, see [Configure the Subject Name Identifier Sent to the Application](#) in the documentation of SAP Cloud Identity Services.

Attribute Mapping in SAP Cloud Identity Services Tokens

User Attribute Expected by SAP BTP	Purpose
Subject name identifier	User identifier for Neo subaccounts. Default value: User ID

User Attribute Expected by SAP BTP	Purpose
mail	<p>E-mail address of the user.</p> <p><b> ⓘ Note</b></p> <p>User identifier for all the account levels (global account, directory, multi-environment subaccount), and for the Cloud Foundry environment.</p> <p>The e-mail addresses of all the users in the SAP Cloud Identity Services tenant must be unique.</p> <p>For more information, see the <a href="#">prerequisites [page 2239]</a> for establishing trust and federation of custom identity providers for platform users.</p>
first_name	First name of the user.
last_name	Last name of the user.
groups	Any groups the subject is assigned to in the identity provider.

## Related Information

[Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#)

[Map User Attributes from a Corporate Identity Provider for Business Users \[page 2208\]](#)

## 6.1.1.8.1.2.2 Supported Tools and Services When Using Custom Identity Providers for Platform Users

Not all tools and services of SAP BTP support the use of custom identity providers with platform users. We provide a list of tools and services, which support this feature and any restrictions that apply.

### Supported When Using Identity Providers for Platform Users

Supported	Description
Cloud Connector	<p>You can connect a Cloud Connector with a multi-environment subaccount of SAP BTP.</p> <p>For more information, see <a href="#">Use a Custom IDP for Subaccount Configuration</a>.</p> <div style="background-color: #f0f0f0; padding: 10px;"><p><b>⚠ Restriction</b></p><p>Currently, Cloud Connector can be connected to an SAP BTP subaccount only with users from the default identity provider.</p></div> <div style="background-color: #f0f0f0; padding: 10px;"><p><b> ⓘ Note</b></p><p>In addition to the initial connection from Cloud Connector to SAP BTP, the regular rotation of the client certificate that is used for the ongoing connection, is also affected.</p></div>
SAP BTP Command Line Interface (btp CLI)	<p>Once trust has been configured on the global account level in the cockpit, platform users can log on using the custom identity provider and use the full range of the btp CLI.</p> <p>To log on to a global account, you need to provide the hostname of the tenant (for example, <code>ar9ibaxhm</code>) with the <code>--idp</code> parameter. See <a href="#">Log in with a Custom Identity Provider [page 2342]</a>.</p> <p>To work with users and role collections on global account, directory, or subaccount level, you need to provide the origin key with the <code>--of-idp</code> parameter in the following commands:<code>btp list security/user</code>,<code>btp get security/user</code>,<code>btp delete security/user</code>,<code>btp assign security/role-collection</code>,<code>btp unassign security/role-collection</code>.</p> <p>Both values can be found in the cockpit under <a href="#">Security → Trust Configuration → Custom Platform Identity Providers</a>.</p>

Supported	Description
Cloud Foundry command-line interface (CF CLI)	<p>Make sure you use the Cloud Foundry command-line interface (CF CLI v7). For more information, see <a href="https://docs.cloudfoundry.org/cf-cli/v7.html">https://docs.cloudfoundry.org/cf-cli/v7.html</a>.</p>
	<p><b>⚠ Restriction</b></p> <p>With v6 of the CF CLI, you can't use the <code>--origin</code> option in <code>cf set-org-role</code> and <code>cf set-space-role</code>. Use the  <code>Members</code> tab in the SAP BTP cockpit. Go to the SAP BTP cockpit under     <code>Members</code> to set the user's permissions.</p>
	<p>For more information, see <a href="#">Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface [page 2251]</a>.</p> <p>See also <a href="#">Using the Cloud Foundry CLI with SAP HANA Cloud</a></p>
SAP Application Logging service and Kibana	<p>Users logged on using custom identity providers can analyze logs written using the SAP Application Logging service.</p> <p>For more information, see the <a href="#">SAP Application Logging service</a> documentation.</p>
SAP BTP cockpit	<p>Platform users logged on using custom identity providers can use the full range of the SAP BTP cockpit.</p>
SAP Cloud Transport Management	<p>Platform users logged on to SAP Cloud Transport Management service using custom identity providers can use all functions of the service as described in the SAP Cloud Transport Management documentation.</p> <p>For more information, see <a href="#">SAP Cloud Transport Management</a>.</p>
SAP HANA Cloud Cockpit	<p>Platform users logged on using custom identity providers can use the full range of SAP HANA Cloud Cockpit.</p> <p>For more information, see <a href="#">SAP HANA Cockpit</a>.</p>
SAP Job Scheduling service	<p>Platform users logged on using custom identity providers can use the SAP Job Scheduling service dashboard.</p>
SAP Mobile Services	<p>SAP Mobile Services uses platform users to log on to the admin cockpit and manage apps and configurations.</p> <p>Based on their space member roles, platform users can log on using a custom identity provider and access the SAP Mobile Services admin cockpit.</p> <p>For more information, see <a href="#">Set Up Customer Accounts</a> in the SAP Mobile Services documentation.</p>

Supported with Restrictions When Using Platform Identity Providers

Supported with Restrictions	Description
SAP Business Application Studio	<p>SAP Business Application Studio supports custom identity providers with the following restrictions.</p> <ul style="list-style-type: none"><li>When you log on with the SAP Business Application Studio web-based user interface, you authenticate with the identity provider for business users.</li><li>When you log on to the Cloud Foundry environment from the terminal mode of SAP Business Application Studio with the Cloud Foundry environment command-line interface (CF CLI), you can authenticate with the identity provider for platform users.</li></ul> <p>For more information, see <a href="#">Terminal</a> in the SAP Business Application Studio documentation.</p>
SAP HANA Service Dashboard	<p>This tool doesn't support log on with a platform user from a custom identity provider. You can only log on with a user from the default identity provider.</p> <ul style="list-style-type: none"><li><a href="#">Open the SAP HANA Service Dashboard</a> in AWS and Google Cloud regions</li><li><a href="#">Open the SAP HANA Service Dashboard</a> in the China (Shanghai) region</li></ul>
SAP Rapid Application Development by Mendix	<p>You can't perform operations on multi-environment subaccounts, like deployments with SAP Rapid Application Development by Mendix when using a custom identity provider for platform users.</p> <p>For more information, see <a href="#">SAP Rapid Application Development by Mendix</a>.</p>
Service dashboards of services that aren't listed as supported.	<p>If you use services with dashboards, you can't access these dashboards with a platform user from a custom identity provider. Using service dashboards is only possible with the default identity provider (SAP ID service).</p>
SAP Web IDE	<p>SAP Web IDE running in a multi-environment subaccount of SAP BTP only supports users from SAP ID service.</p> <p>For more information, see <a href="#">User Authentication and Authorization</a>.</p>

#### → Recommendation

Use SAP Business Application Studio.

## Related Information

[Administration and Operations in the Cloud Foundry Environment \[page 2422\]](#)

## 6.1.1.8.1.2.3 Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface

Learn how to use different methods to log on to Cloud Foundry using a custom identity provider (IdP).

### Prerequisites

- You've created at least one subaccount and enabled the Cloud Foundry environment in this subaccount.  
For more information, see [Create a Subaccount \[page 2173\]](#).
- You've downloaded and installed the Cloud Foundry command-line interface (cf CLI).  
For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).
- Your administrator has configured your Cloud Foundry environment to use a custom identity provider.  
For more information, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).
- You've configured the login screen to display your custom identity provider, see [Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#) (relevant for the manual login process).

### Context

The cf CLI provides different options to log on using a custom identity provider. One scenario is recommended for an interactive logon because you must switch from the CLI to the browser to log on. This scenario supports multifactor authentication and other passwordless authentication methods, as well as single sign-on. The other scenario focuses on automation scenarios where switching between the CLI and a browser isn't possible.

### Procedure

Decide which scenario applies to you according to this table.

Scenario	See
You can open a browser during the logon process.	<a href="#">Log On Manually With a Custom Identity Provider [page 2252]</a> .
The logon process is automated, for example with a script or there's no possibility to open a browser during logon.	<a href="#">Log On as a Technical User With a Custom Identity Provider [page 2254]</a>

Scenario	See
<p><b>⚠ Restriction</b></p> <p>This scenario is only supported if the users exist directly in your tenant of SAP Cloud Identity Services and not in a corporate identity provider.</p>	

## Related Information

[Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#)

### 6.1.1.8.1.2.3.1 Log On Manually With a Custom Identity Provider

To log on to Cloud Foundry, using a custom identity provider, use the single sign-on option of the CF CLI.

#### Prerequisites

- You've configured the login screen to display your custom identity provider.  
For more information, see [Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#).
- You know the origin key of the identity provider.  
For more information, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).

#### Context

During logon, the system provides you with a URL to your custom identity provider for you to enter in a web browser. If you're already logged on to your custom identity provider, the system provides a temporary authentication code to complete your logon. If you don't have an active session with your custom identity provider, you need to log on to receive the temporary authentication code.

## Procedure

1. Open a command line.
2. Set the target API endpoint to endpoint of your subaccount.

```
cf api https://api.cf.<region>.hana.ondemand.com
```

### ↔ Sample Code

```
cf api https://api.cf.eu10.hana.ondemand.com
```

3. Log on to the Cloud Foundry environment with single sign-on:

```
cf login --sso
```

The command-line interface displays a login URL and prompts you to enter a temporary authentication code.

4. Open the URL that is provided on the screen to go to the logon screen.

If this is the first time you're logging on or your browser cache has been cleaned of cookies, you're prompted to provide the origin key of your custom identity provider or choose the default identity provider.

Otherwise, the sign-in page lists your accounts on custom identity providers with which you previously logged on.

5. Choose the account of your identity provider.

If you don't have an active session yet, you're prompted to log on to your identity provider.

After you've successfully logged on, you receive a temporary authentication code.

6. Enter the passcode.

You've logged on successfully. You can now choose your org.

## Related Information

[Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#)

## 6.1.1.8.1.2.3.2 Log On as a Technical User With a Custom Identity Provider

To log on to Cloud Foundry, using a custom identity provider, use the `--origin` option of the Cloud Foundry command-line interface (cf CLI).

### Prerequisites

The users exist:

- Directly in your tenant of the SAP Cloud Identity Services.
- In a corporate identity provider with SAP Cloud Identity Services working as a proxy.
  - Trust to your SAP Cloud Identity Services tenant must be configured with OIDC and not SAML.
  - Your corporate identity provider must support the password grant flow.

For more information, see [Configure Trust with OpenID Connect Corporate Identity Provider](#) in the documentation for SAP Cloud Identity Services.

### Context

We recommend this method of logging on if you want to use an automated script and can't open a browser during the logon process.

### Procedure

Set up a script with the following code:

```
cf api https://api.cf.<region>.hana.ondemand.com  
cf login --origin <origin> -u <user> -p <password>
```

#### ↳ Sample Code

```
cf api https://api.cf.eu10.hana.ondemand.com  
cf login --origin sap.ids -u julie.armstrong@sap.com -p mysecurepassword
```

- Find the `<region>` value, that applies to you in the section [Regions \[page 17\]](#).
- Find the `<origin>` value for the user that you want to use in the cockpit.
  1. Navigate to the subaccount of your user.
  2. Choose  .
  3. Find the `<origin>` value of your user in the table.

## 6.1.1.8.1.2.4 Log On with a Custom Identity Provider to the SAP BTP Cockpit

All users can log on to the SAP BTP cockpit with a custom identity provider.

### Prerequisites

- Your user has administration or viewer rights in this subaccount or global account. For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).
- You have established trust with a custom identity provider for platform users. For more information, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).

### Context

Use the [Open](#) link in the *SAP BTP Cockpit* column. It contains a URL for the user to log on with the custom identity provider.

#### Example

<https://emea.cockpit.btp.cloud.sap/cockpit/?idp=cidppuxhm.accounts.ondemand.com>

### Procedure

1. Go to your global account or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose ► **Security** ➤ **Trust Configuration** ➤ in the SAP BTP cockpit.
2. Use the [Open](#) link in the SAP BTP *Cockpit* column.
  - To enable other platform users to log on with a custom identity provider, send the link to these users.
  - To work with your user accounts from multiple identity providers at the same time, open the link in a private or incognito browser window or tab.

#### ⓘ Note

A browser that isn't in private or incognito mode has one single session with the cockpit. If you have an existing cockpit session with one identity provider and log on with another one, the existing session closes and creates a new session. For this reason, you can't have parallel sessions with different identity providers.

3. Log on using the custom identity provider.

## 6.1.1.8.1.2.5 Log on with a Browser to the Cloud Foundry CLI and Service Dashboards

Platform users of the Cloud Foundry environment have the option to log on with a custom identity provider or the default identity provider.

### Prerequisites

- You know the origin key of the identity provider.  
For more information, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).
- You don't already have a session with the Cloud Foundry environment.  
If you're already authenticated, you skip the logon process and go straight to your application.

### Context

When accessing a web application, like a service dashboard (for example `logs.cf.eu10.hana.ondemand.com`) or you log into your Cloud Foundry account using the CLI with `cf login -sso`, you're prompted by a login screen. This method enables you to log on to a custom identity provider or makes it easier to log on when multifactor authentication is required.

### Procedure

1. Navigate to the URL for your service dashboard or provided by the CF CLI.

The login screen appears.

If this is the first time you're logging on or your browser cache has been cleaned of cookies, you're prompted to provide the origin key of your custom identity provider or choose the default identity provider.

# SAP Business Technology Platform

Choose your identity provider

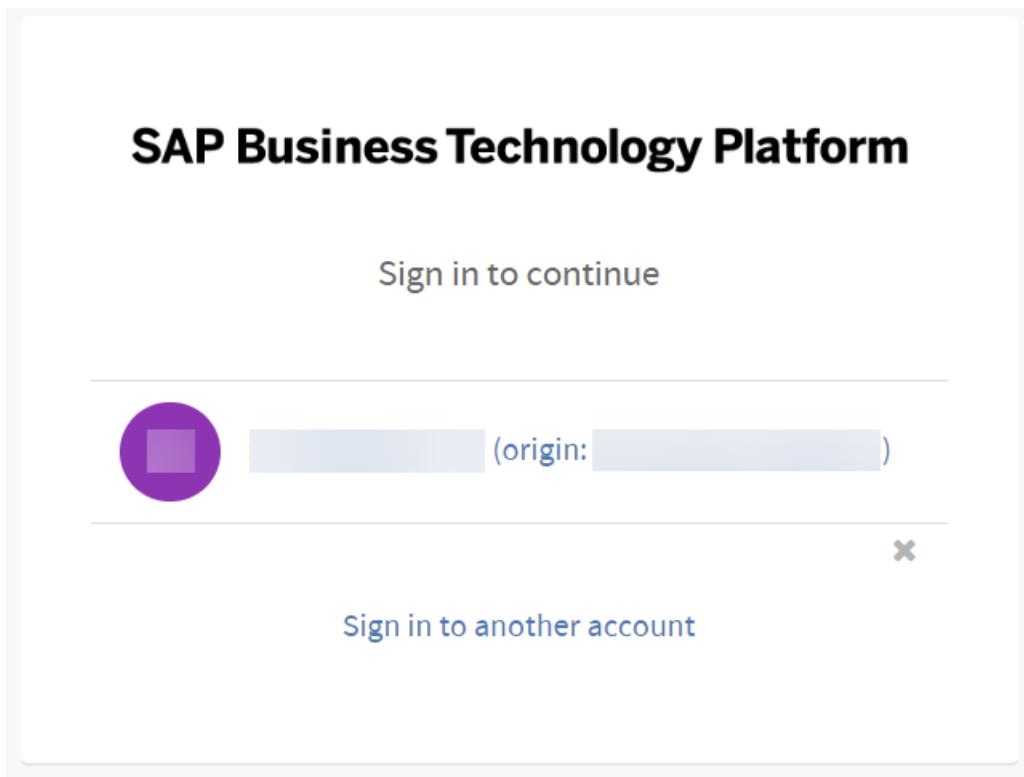
**Sign in with default identity provider**

Enter the origin key of your identity provider

**Sign in with alternative identity provider**

[Logon Page Without Cookies](#)

Otherwise, the sign-in page lists your accounts on custom identity providers with which you previously logged on.



Logon Page With Previous Logon

2. Choose your identity provider and log on.

## Related Information

[Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command-Line Interface \[page 2251\]](#)

### 6.1.1.8.1.3 Default Identity Provider

SAP ID service is the default identity provider for both platform users and business users (in applications) at SAP BTP. You can start using it without further configuration.

#### ⓘ Note

For China (Shanghai) region, a different default identity provider is used.

For more information, see this [blog article](#) on SAP Community.

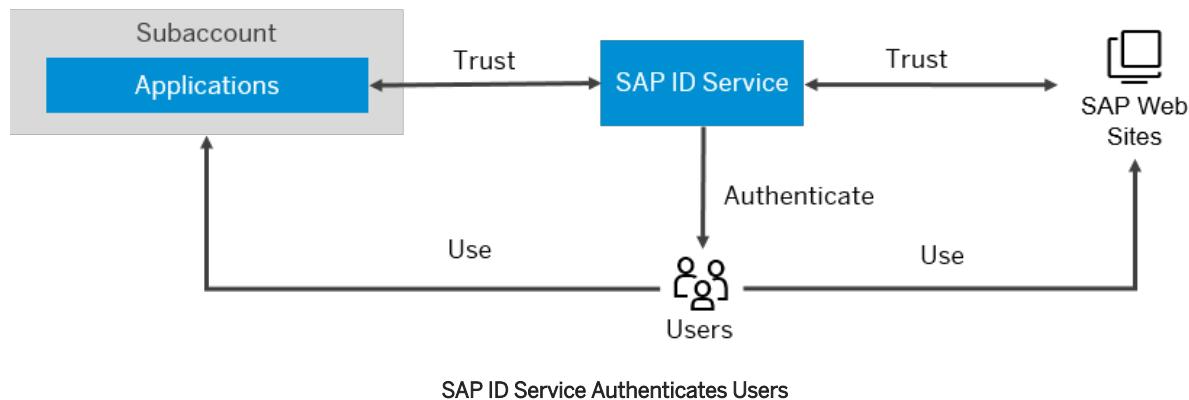
For Government Cloud (US) region, a different default identity provider is used.

SAP ID service provides:

- A central user store

- A Single Sign-On (SSO) service. It enables users to log on once and get access to all your applications.

Use SAP ID service as a preconfigured user store in your starter scenarios or for testing. You can also use the default identity provider as a backup identity provider if access to your custom identity provider fails. SAP ID service is the place where you register to get initial access to SAP BTP.



If you're a new user, you can use the self-service registration option at the [SAP Web site](#). The SAP website registers you with SAP Universal ID, which also registers you with SAP ID service.

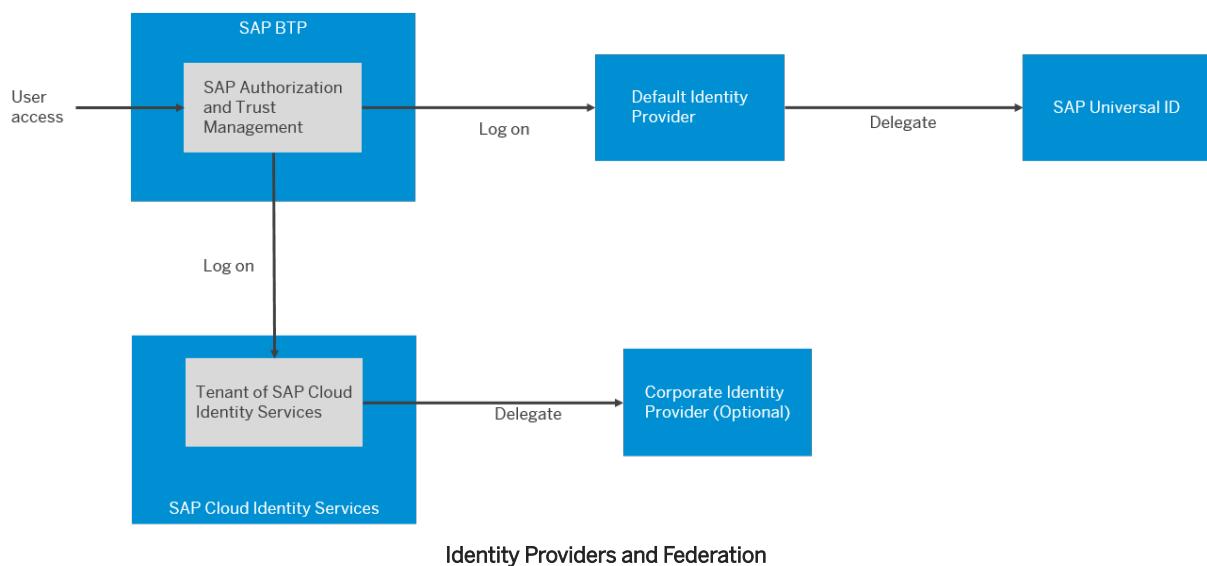
## SAP Universal ID

### i Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

SAP Universal ID manages the users of official SAP sites, including the SAP developer and partner community. If you already have such a user, then you're already registered with SAP ID service as well. SAP ID service acts as a proxy for SAP Universal ID, when users log on with their e-mail addresses. Users can log on with and manage all their user accounts with SAP Universal ID.

The following figure illustrates how default and custom identity providers can federate other identity providers.



## Managing Users

To add users to a subaccount, the users must exist in an identity provider.

For more information about adding users to our default identity providers, see [Create SAP User Accounts \[page 2262\]](#).

To add new users to a subscribed app or service, such as Web IDE, add those users to your subaccount.

For more information, see [Add Users from SAP ID Service for Multi-Environment Subaccounts \[page 2263\]](#).

## Multifactor Authentication

For SAP ID service, you can ask users hosted by this identity provider to enable multifactor authentication as a self-service. There's **no** mechanism to enforce users in SAP ID service to use this function.

### → Recommendation

To enforce multifactor authentication, we recommend that you trust SAP Cloud Identity Services - Identity Authentication and configure enforcement. Optionally, integrate SAP Cloud Identity Services with your corporate identity provider and enforce the policy there.

For more information, see:

- [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#)
- [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#)

For more information about your users of SAP ID service can enable multifactor authentication themselves, see [How to Enable Multi-Factor Authentication \(MFA\) 🛡️](#) on the SAP Support Portal.

### ⓘ Note

Some user interfaces don't offer an interactive way to support multifactor authentication, such as time-based one time passwords (TOTP). Instead, such tools offer fixed logon ID and password fields. For such

tools, enter your password directly followed, without any spaces or dividers, by the TOTP offered by your multifactor device.

User ID: **MylogonID**

Password: **MystrongpasswordMytotppasscode**

For some tools, this behavior affects log on to the tool itself:

- Cloud Foundry command-line interface (cf CLI)  
Alternatively, you can log on through a browser instead.  
For more information, see [Log On Manually With a Custom Identity Provider \[page 2252\]](#).
- SAP Business Technology Platform command-line interface (btp CLI)  
Alternatively, you can log on through a browser instead.  
For more information, see [Log in with Single Sign-On \[page 2341\]](#).

For other tools, this behavior doesn't affect log on to the tool itself, but log on to the platform when establishing connections or deploying software and such. For example:

- Cloud Connector
- SAP Business Application Studio
- SAP Web IDE

## Trust and Identity Providers

Trust between your subaccount and SAP ID service is preconfigured by default, so you can start using it without further configuration.

You can hide the default trust.

For more information, see [Hide Logon Link for Default Identity Provider \[page 2235\]](#).

To use a custom identity provider, establish trust to your custom identity provider. We describe a custom trust configuration using the example of SAP Cloud Identity Services.

For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

## Related Information

[Security \[page 3021\]](#)

[Platform Identity Provider for Tools](#)

[Working with Users \[page 2265\]](#)

## 6.1.1.8.1.3.1 Create SAP User Accounts

To grant authorizations to people from the default identity provider in your subaccount, ensure that they have a user account.

### Context

If the person in question already has a user account on `sap.com` Web sites, then that person already has a user in the default identity provider. Add this user to your accounts, if you know this person's e-mail address.

Consider using a custom identity provider to integrate your own identity and access management solution.

#### ⓘ Note

For technical users, create the user in your custom identity provider. In your custom identity provider, you control the access policies of these users independently from the policies we set for the default identity provider. Our default identity provider is meant for **human** users, who interact with SAP and our solutions.

### Procedure

Send your colleagues the self-registration URL.

```
https://account.sap.com/core/create/register?  
redirectURL=https%3A%2F%2Femea.cockpit.btp.cloud.sap%2Fcockpit%2F
```

The Web site registers you with SAP Universal ID, which also registers you with the SAP ID service. If you already have a user in the SAP ID service, associate this user with your new SAP Universal ID account. SAP Universal ID manages the users of official SAP sites, including the SAP developer and partner community.

#### → Tip

After your colleagues sign up, add them to a subaccount. Assign role collections to them. Then your colleagues can log on to the SAP BTP cockpit.

### Related Information

[Add Users from SAP ID Service for Multi-Environment Subaccounts \[page 2263\]](#)

[Default Identity Provider \[page 2258\]](#)

[Trust and Federation with Identity Providers \[page 2204\]](#)

## 6.1.1.8.1.3.2 Add Users from SAP ID Service for Multi-Environment Subaccounts

Before you can assign role collection to a user from SAP ID service, ensure that this user exists in your subaccount.

### Prerequisites

The user you want to add to your subaccount must have an SAP user account (for example, an S-user or P-user).

For more information, see [Create SAP User Accounts \[page 2262\]](#).

### Context

When you create your own trial account, your SAP user is automatically created and assigned to SAP ID service. When you onboard new members to your subscribed app, add them to your subaccount. Then you can assign a role collection to the user.

### Procedure

Create users and assign role collections to users in ► [Security > Users](#) ▾.

It's possible to create users in custom identity providers and in the SAP ID service.

For more information, see [Working with Users \[page 2265\]](#).

### Related Information

[Platform Identity Provider for Tools](#)

[Mapping Role Collections in the Subaccount \[page 2281\]](#)

## 6.1.1.8.1.4 Switch Off Automatic Creation of Shadow Users

To switch off the creation of shadow users in the trust configuration of custom identity providers, administrators must explicitly allow users to log on. Administrators then have full control over who is allowed to log on.

### Context

Whenever a user authenticates at an application in your subaccount using any identity provider, the User Account and Authentication service always stores user-related data provided by the identity provider in the form of shadow users. The UAA uses details of the shadow users to issue tokens that refer to a certain user.

By default, the UAA allows any user of any connected identity provider to authenticate to applications in the subaccount. When there's no corresponding shadow user, it automatically creates one based on the information received from the identity provider.

#### Note

With new subaccounts created after 24 September 2020, automatic creation of shadow users is switched off by default for the default identity provider, SAP ID service.

Usually, you want your administrators to be fully aware of which users they allow to log on. If you've switched off automatic creation of shadow users for a certain identity provider, you enforce that only those users can log on where shadow users have been created explicitly. You can create them in the SAP BTP cockpit, typically when you assign the first role collection to them. For more information, see [Create Users \[page 2270\]](#).

To switch off creation of shadow users during logon, proceed as follows:

### Procedure

1. In the SAP BTP cockpit, go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose  [Security](#)  [Trust Configuration](#) .
2. Choose your custom trust configuration.
3. Choose [Edit](#).
4. Go to [Create Shadow Users on User Logon](#) and remove the checkmark.
5. Save your changes.

From now on, the User Account and Authentication service doesn't automatically create shadow users for a certain identity provider during logon.

### Related Information

[Deletion \[page 3114\]](#)

[Delete Users \[page 2272\]](#)

[Delete Shadow Users for Data Protection and Privacy Using APIs \[page 3118\]](#)

## 6.1.1.8.2 Working with Users

In the SAP BTP cockpit, you can see the users of your global account or subaccount, user-related identity provider information, and their authorizations. In a user's overview, you can create and delete users, and assign role collections. You can also display an overview of the role collections, where you can drill down all the way to the role, and see the application that the role belongs to.

### Overview

All users in the global account, subaccount, directories, and environments are stored in identity providers. Users in the subaccounts of SAP BTP are either stored in the default identity provider or in a custom identity provider.

SAP BTP creates a copy of the user in the global account or in the subaccount when a user-related action happens. This copy of the user is called a shadow user. Global account or subaccount administrators of SAP BTP assign role collections to shadow users. To comply with data protection and privacy regulations, administrators can be obliged to delete shadow users who belonged to employees who left the company.

#### → Remember

For access tokens of subaccounts created after September 23, 2020, user IDs preserve the case that they are created in. This means that both uppercase and lowercase letters can be used. However, this does not qualify as a means to differentiate between them; for example, denise.smith@example.com and Denise.Smith@example.com are considered to be the same.

Subaccounts before this date convert the user IDs to lowercase.

#### ⓘ Note

User data of shadow users is regularly updated after authentication at the identity provider.

When a user authenticates against SAP BTP, an authentication request is forwarded to the trusted identity provider (for example, SAP Cloud Identity Services). After successful authentication, the identity provider creates a user token based on its configuration and sends it back to SAP BTP. In SAP BTP, the user data is updated according to the token.

### Prerequisites

- You have administration rights in this global account, subaccount, or directory. For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

- The users are stored in identity providers that are connected to SAP BTP.
  - Default identity provider. For more information, see [Default Identity Provider \[page 2258\]](#).
  - Custom identity provider. For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

Use Case	Description
Enable a new hire to log on to the subaccount and have access to necessary resources.	<p>If automatic creation of shadow users was switched off for this identity provider, shadow users aren't created automatically in the subaccount (see the related link). This means that the new user can't log on to the subaccount. As an administrator, create a shadow user that corresponds to the user in the identity provider and assign the necessary role collections to this user.</p> <p>See <a href="#">Create Users [page 2270]</a>.</p>
	<p>If automatic creation of shadow users was switched off for this identity provider, shadow users aren't created automatically in the subaccount (see the related link). Add the user group to the role collections that are necessary for this user if the user belongs to a user group in the identity provider.</p> <p>See <a href="#">Assign User Groups to Role Collections [page 2279]</a>.</p>
Enable a new hire to have access to necessary resources.	<p>A shadow user is automatically created in the subaccount. This means that the new user can log on to the subaccount, but the user doesn't have any authorizations. As an administrator, you have created new role collections or you want to use existing ones for the new hire. You grant authorizations by assigning the role collections to this user.</p> <p>See <a href="#">Assign Users to Role Collections [page 2278]</a>.</p>
Enable a new hire to log on.	<p>If automatic creation of shadow users was switched off for this identity provider, shadow users aren't created automatically in the subaccount (see the related link). As an administrator, create a new shadow user in the subaccount.</p> <p>See <a href="#">Create Users [page 2270]</a>.</p> <p>If your subaccount uses the default trust configuration, you can also create a shadow user there.</p> <p>See <a href="#">Add Users from SAP ID Service for Multi-Environment Subaccounts [page 2263]</a>.</p>

Use Case	Description
Review authorizations of individual users.	<p>You want to find out which authorizations have been granted to users. To do so, access the list of all users, find the users you want to review, and check the role collection assignments. From the user's role collection, you can drill down to the roles of the role collection. You can also assign and remove role collections.</p> <p>See <a href="#">Find Users and Their Role Collection Assignments [page 2268]</a>.</p>
Delete users of employees who have left the company.	<p>You want to find and delete users of employees who have left the company, but their users still exist in the system.</p> <p>How to find users, see <a href="#">Find Inactive Users [page 2268]</a>.</p> <p>How to delete users, see <a href="#">Delete Users [page 2272]</a>.</p>
Automate the provisioning of users.	<p>You want to provision users to SAP cloud solutions, such as SAP BTP so that you can manage your identities centrally.</p> <p>How to provision users to SAP BTP, see <a href="#">Provision Users to SAP BTP Accounts [page 2271]</a>.</p>
Find users who have never logged on.	<p>You want to find users who never logged on. For example, they could be manually created shadow users with a typo in their user ID, so they don't match the user ID provided by the identity provider. They were never used, and it might make sense to delete them.</p> <p>See <a href="#">Delete Users [page 2272]</a>.</p>
Delete shadows user to comply with data protection and privacy regulations.	<p>For data protection and privacy reasons, administrators might be legally obliged to delete shadow users who belonged to employees who left the company. You can find and delete the shadow users in the SAP BTP cockpit.</p> <p>See <a href="#">Delete Users [page 2272]</a>.</p>
	<p>You can also delete the users using APIs.</p> <p>See <a href="#">Delete Shadow Users for Data Protection and Privacy Using APIs [page 3118]</a>.</p>

## Related Information

[Switch Off Automatic Creation of Shadow Users \[page 2264\]](#)

[Security Administration: Managing Authentication and Authorization \[page 2202\]](#)

[Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)

## 6.1.1.8.2.1 Find Users and Their Role Collection Assignments

You want to find the authorizations granted to specific users in your global account and subaccount. You can search or sort the list of users, and view their role collection assignments.

### Context

You want to find out which authorizations have been granted to single or multiple users. To do so, access the list of all users, find the users you want to review, and check the role collection assignment. From the user's role collection, you can drill down to the roles of the role collection.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and/or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  .

You see all the users of your subaccount.

→ Tip

If you want to see users that were updated before a certain point of time, use the *Last Updated* field. You can filter the list according to the point of time of the last update.

4. To search for users, choose .
5. Enter a string occurring in the user name, last name, first name, or e-mail address. The SAP BTP cockpit returns a list of all the users containing this string.
6. Select a user and open the overview section.

You can see all of the role collection assignments of this user. You can also drill down into the roles.

## 6.1.1.8.2.2 Find Inactive Users

As an administrator, you want to have an overview of the users in your subaccount, no matter which identity provider stores them. You want to clean them up if necessary.

### Context

- You want to find users that haven't been used for some time. The reason could be that employees left the company, but the users still exist in the system.

- You want to find users who never logged on. For example, they could be manually created shadow users with a typo in their e-mail address, so they didn't match the user ID provided by the identity provider. They were never used, and it might make sense to delete them.

## Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and/or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose [Users](#).

You see all the users in your subaccount.

4. If you want to see users that were updated at a certain point in time, use the *Last Updated* field. You can filter the list according to the following filter criteria:

- All
- Today
- This week
- This month
- This year

If you want to see the users who haven't logged on at a certain point in time, use the *Hasn't Logged On* field. You can filter the list according to the following filter criteria:

- All
- Last 7 Days
- Last 30 Days
- Last 180 Days
- Last 365 Days
- Choose a Date

The SAP BTP cockpit displays the filtered list of users. The *Last Updated* column shows you the point in time when the user was updated last. The *Last Logon* column shows you the point of time of the user's last logon.

5. (Optional) To see whether a user has never logged on, choose the user to display the user's overview section.

The SAP BTP cockpit displays the user's overview section. If the user has never logged on, the *Last Logon* field shows the message *User has never logged on*.

6. (Optional) To delete the user, close the overview section and choose (Delete).

### → Remember

You can't undo the deletion of a user.

## 6.1.1.8.2.3 Create Users

As an administrator, you can create shadow users in your subaccount. When you create a shadow user, you must know and specify which identity provider stores the user.

### Prerequisites

You've administration rights in this subaccount. For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

### Context

All users in the subaccounts of SAP BTP are stored in identity providers, either in the default or in a custom identity provider. SAP BTP creates a copy of the user in the subaccount when a user-related action happens. The copy of the user in the subaccount is called shadow user. It's also possible to switch off the creation of shadows users (see the related link).

A new hire starts to work in your company next month. As an administrator, you want to provide a user, enable logon to the SAP BTP subaccount, and provide the necessary authorizations for this new hire. You create the actual user in the identity provider. However, you can't assign subaccount authorizations to an individual user in an identity provider. You need a shadow user in the SAP BTP subaccount to assign authorizations. After having created a shadow user, you can assign role collections. Thus, you make sure that the new hire can log on to the subaccount and that the necessary authorizations are in place when the new hire starts.

If you switched off the creation of shadow users in the trust configuration of your custom identity providers, new users created in the identity provider don't have a user copy (shadow user) in the respective SAP BTP subaccount. You, as a subaccount administrator, have full control of who can log on to this subaccount. The new user can log on to the subaccount if you've created a shadow user with the same email address in the SAP BTP subaccount. Now, you can grant authorizations by assigning role collections.

You can alternatively grant authorizations in the subaccount by mapping role collections to the user group if the user belongs to a user group in the identity provider.

As an administrator, you can create shadow users in your subaccount and you must determine which identity provider stores the user. You can then give the user authorizations by assigning role collections to the user.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and/or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  **Security**  **Users** .
4. Choose [Create](#).

The SAP BTP cockpit displays a new row where you can enter the user data.

5. Enter the user ID and email address.

**ⓘ Note**

The email address of the user is the user identifier for all account levels (global account, directory, multi-environment subaccount) and for the Cloud Foundry environment.

**ⓘ Note**

To specify the last name and first name of the user, maintain the names in the identity provider that stores the user. This information is automatically added to the shadow users when the user logs on with their identity provider using a URL such as:

`https://<myapplication>.cfapps.<region>.hana.ondemand.com/`

For example: `https://my-application.cfapps.eu10.hana.ondemand.com/`

Logging on with the SAP BTP cockpit or the command-line interface isn't enough.

6. Choose the identity provider where the user is stored. The dropdown list displays the identity providers configured in the trust configuration of your subaccount.
7. Save your changes.

You can now proceed to assign role collections to the new user.

## Related Information

[Switch Off Automatic Creation of Shadow Users \[page 2264\]](#)

### 6.1.1.8.2.4 Provision Users to SAP BTP Accounts

As people join, change roles, and eventually leave your organization, you care for the onboarding, maintenance, and offboarding of their users in your systems. If you're a small organization, you can use manual processes; otherwise, you should consider automating your processes.

## Prerequisites

- You have the required authorizations.  
For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).
- For global account, directory, and subaccount levels, you have API credentials.  
For more information, see [Accessing Administration Using APIs of the SAP Authorization and Trust Management Service \[page 2416\]](#).

## Context

We recommend creating users within the SAP Cloud Identity Services. The persistency layer of SAP Cloud Identity Services is the Identity Directory. Use SAP Cloud Identity Services - Identity Provisioning to provision users to SAP cloud solutions, such as SAP BTP. Identity Provisioning has connectors to distribute user artifacts needed at different levels of your SAP BTP accounts. By managing your identities centrally, you also ensure that when you offboard users, they're removed from target systems.

## Procedure

Integrate with your identity and access management solution.

- To integrate with SAP Cloud Identity Services, see [SAP BTP Integration Scenario](#) for more information.
- To integrate with another solution, use the User Management SCIM API. See [User Management \(System for Cross-domain Identity Management \(SCIM\)\)](#) for more information.

### ⓘ Note

For user management of Cloud Foundry organizations, use the provisioning connectors of SAP Cloud Identity Services - Identity Provisioning. For more information, see [SAP BTP Integration Scenario](#).

### 6.1.1.8.2.5 Delete Users

As an administrator, you can delete users from your subaccount. When you delete a user, you also delete the user's role collection assignments.

## Prerequisites

You have the required authorizations.

For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

## Context

All users in the subaccounts of SAP BTP are shadow users. As an administrator, you might want to clean up users from your subaccount. Or for data protection and privacy reasons, it might be a legal obligation that administrators delete shadow users who belonged to employees who left the company. For more information, see [Working with Users \[page 2265\]](#).

It's also possible to delete users using APIs. For more information, see the related links.

Keep in mind that you also delete the user's role collection assignments.

#### → Remember

When you delete a user at the Cloud Foundry level, SAP BTP doesn't delete the corresponding shadow user at the subaccount level.

When you delete a user at the subaccount level, ensure that you delete the user at the Cloud Foundry level too. Otherwise, if the Cloud Foundry user logs on again, the system automatically creates the corresponding user at the subaccount level.

So ensure that you delete the shadow users on all levels.

#### ⚠ Caution

You cannot undo the deletion of a user.

## Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account, directory, or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  

#### ⓘ Note

For directories, choose [Users](#). There is no [Security](#) menu item.

4. Choose from the following options:

Options	Procedure
<b>Delete a single user</b>	<ol style="list-style-type: none"><li>1. Select the row of a user.</li><li>2. Choose .</li></ol>
<b>Delete multiple users</b>	<ol style="list-style-type: none"><li>1. Choose  <a href="#">Select multiple users</a> and select users (see the related link).</li><li>2. Choose <a href="#">Delete</a>.</li></ol>

## Related Information

[Working with Users \[page 2265\]](#)

[Deletion \[page 3114\]](#)

[Delete Shadow Users for Data Protection and Privacy Using APIs \[page 3118\]](#)

### 6.1.1.8.3 Working with Role Collections

As an administrator, you group application roles in role collections. You then assign role collections to application users.

Application developers have defined application-specific role templates in the security descriptor file. The role templates contain the role definitions. You can assign the role to a role collection.

#### → Tip

Application developers can even directly define role collections with default roles. For more information, see [Create Role Collections with Predefined Roles \[page 537\]](#).

Typically, these role collections provide authorizations for certain types of users, for example, sales representatives.

Once you have created a role collection, you can pick the roles that apply to the typical job of a sales representative. Since the roles are application-based, you must reference the application to see which roles come with the role template of this application. You are free to add roles from multiple applications to your role collection.

Finally, you assign the role collection directly to users or indirectly to attributes such as groups. For more information, see the related link.

## Prerequisites

- Your user has administration rights in this subaccount and or global account. For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).
- The roles defined by your application developers in the application security descriptor are available in the SAP BTP cockpit. For more information, see [Adding Authentication and Authorization \[page 499\]](#).
- The users are stored in identity providers that are connected to SAP BTP.
  - Default identity provider (SAP ID service). For more information, see [Default Identity Provider \[page 2258\]](#).
  - Custom identity provider. For more information, see [User Management](#) in SAP Cloud Identity Services.

#### ⓘ Note

SAP BTP provides default role collections. You can use the default role collections, but you can't change or delete them. For this reason, the **Delete** icon of a default role collection is grayed out.

You can perform the following tasks:

- [Define a Role Collection \[page 2275\]](#)
  - [Add Roles to a Role Collection \[page 2276\]](#)
  - [Delete Roles from a Role Collection \[page 2277\]](#)
- [Assigning Role Collections to Users or User Groups \[page 2277\]](#)
  - [Assign Users to Role Collections \[page 2278\]](#)

- [Delete Users from Role Collections \[page 2279\]](#)
- [Assign User Groups to Role Collections \[page 2279\]](#)
- [Delete User Groups from Role Collections \[page 2280\]](#)

## Related Information

[Security Administration: Managing Authentication and Authorization \[page 2202\]](#)

[Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

### 6.1.1.8.3.1 Define a Role Collection

To define a role collection, you can create a new role collection or copy an existing one. Add the roles, users, and user groups as needed.

#### Context

One way to define a role collection is to create a new role collection. After you've entered a name and description, you find the new role collection in the list of role collections. You can then add the roles you need and assign users and user groups.

If you have an existing role collection that you want to use as a template, it's a good idea to copy it. The copied role collection includes all of the roles of the origin. However, it doesn't include the users or user groups. You can give it a new name and description. You can find the copy in the list of role collections and assign users and user groups.

#### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose **Security** **Role Collections** .
4. To create a new role collection, choose **+** (Create New Role Collection). To copy an existing role collection, choose **Copy** at the end of the row.
5. Enter a new name and description. If you copied an existing role collection, you can see the included roles.
6. Save your changes.

You can now add or remove roles and assign users or user groups.

## 6.1.1.8.3.1.1 Add Roles to a Role Collection

You can add roles to a role collection.

### Context

The roles are derived from role templates that are provided by applications. For more information, see the related links.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  .
4. To add roles, choose the role collection name.
5. Go to the *Roles* section and choose *Edit*.
6. To add a role to the role collection, choose the input field. The role selection screen opens.

If you accidentally deleted the input field for the new roles, choose  (Add a role) to display a new input field.

7. Use the dropdown list or the F4 function key under *Role Name* to display the roles that are available. The roles are sorted alphabetically. Choose the role you want to add. You can also select multiple roles from the list below.
8. Choose *Add*.
9. Save your changes.

### Related Information

[Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)

[Protecting Your Application \[page 501\]](#)

## 6.1.1.8.3.1.2 Delete Roles from a Role Collection

You can delete roles from a role collection.

### Context

The roles are derived from role templates that are provided by applications. For more information, see the related links.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  .
4. To delete roles, choose the role collection name.  
To drill down into the role details, choose the chevron on the right side of the row.
5. Choose [\*Edit\*](#).
6. To delete a role from the role collection, choose  (Delete) in the row of the role that you want to delete.  
The roles are sorted alphabetically.
7. Confirm the popup.
8. Save your changes.
9. Repeat to delete more roles.

## 6.1.1.8.3.2 Assigning Role Collections to Users or User Groups

You can assign role collections to users and user groups.

### Prerequisites

- You have administration rights in this subaccount or global account. For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).
- The roles defined by your application developers in the application security descriptor are available in the SAP BTP cockpit. For more information, see [Adding Authentication and Authorization \[page 499\]](#).
- The users are stored in identity providers that are connected to SAP BTP.

- Default identity provider (SAP ID service). For more information, see [Default Identity Provider \[page 2258\]](#).
- Custom identity provider. For more information, see [User Management](#) in SAP Cloud Identity Services.

You perform the following tasks to assign role collections to users and user groups stored by identity providers. The identity providers are connected to SAP BTP using trust configurations.

#### Note

Custom identity providers can provide user groups and users whereas default identity providers can only provide individual users.

- [Assign Users to Role Collections \[page 2278\]](#)
- [Delete Users from Role Collections \[page 2279\]](#)
- [Assign User Groups to Role Collections \[page 2279\]](#)
- [Delete User Groups from Role Collections \[page 2280\]](#)

### 6.1.1.8.3.2.1 Assign Users to Role Collections

You can assign users to a role collection by adding them to the role collection.

#### Context

You can assign users from default identity providers, and from custom identity providers, to a role collection. After having entered the user's user ID, choose the origin key of the identity provider and the e-mail address.

#### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose   **Role Collections**.
4. Choose the role collection to which you want to assign users.
5. Go to the **Users** section and choose **Edit**.
6. Enter the user ID, identity provider, and e-mail address of the user that you want to assign to the role collection.
7. (Optional) To add more users, choose  (Add a user).
8. Save your changes.

You've now assigned this user to the role collection. The user has the authorizations of the role collection.

## 6.1.1.8.3.2.2 Delete Users from Role Collections

You can unassign users from a role collection by deleting them from the role collection.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose .
4. Choose the role collection from which you want to unassign users.
5. Choose *Edit*.
6. Choose (Delete) in the row of the user you want to unassign.
7. Save your changes.

You've now unassigned this user by deleting the user from the role collection.

## 6.1.1.8.3.2.3 Assign User Groups to Role Collections

You can assign users groups to a role collection by adding them to the role collection.

### Prerequisites

- You're using a custom identity provider. For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

### Context

You can assign user groups from custom identity providers to a role collection. After having chosen the identity provider, enter the user group name.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).

3. Choose **Role Collections**.
4. Choose the role collection to which you want to assign user groups.
5. Go to the *User Groups* section and choose *Edit*.
6. Select the identity provider where the user group is stored.
7. Enter the name of the user group.

**Note**

If you're using SAP Cloud Identity Services, you can find the name of the user group in the administration console under **Users & Authorizations** **User Groups**. For more information, see the related link.

8. (Optional) To add other user groups, choose (Add a user group).
9. Save your changes.

## Related Information

[User groups in the SAP Cloud Identity Services](#)

### 6.1.1.8.3.2.4 Delete User Groups from Role Collections

You can unassign users groups from a role collection by deleting them from the role collection.

#### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose **Role Collections**.
4. Choose the role collection from which you want to unassign a user group.
5. Go to the *User Groups* section and choose (Delete).
6. Confirm the popup and save your changes.

### 6.1.1.8.3.3 Mapping Role Collections in the Subaccount

You've arranged roles in role collections, and now want to assign or map these role collections to business users.

How you assign users to their authorizations depends on the type of trust configuration and on whether or not you prefer to maintain the authorizations of individual users rather in the identity provider or in SAP BTP. The following options are available:

- Directly assign role collections to users.
- Map role collections to user groups or other user attributes defined in the identity provider. You initially maintain the mapping between user groups or other user attributes and role collections once in SAP BTP, and maintain group memberships or other attributes of users in the identity provider.

#### ⓘ Note

If you're using the default trust configuration with the default identity provider, you directly assign users to role collections. For more information, see [Default Identity Provider \[page 2258\]](#).

However, if you're using a custom trust configuration, for example, with SAP Cloud Identity Services, you can use both options. For more information about configuring the trust between your subaccount and a custom identity provider, see [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#).

Options for Assignment of Role Collections

Trust Configuration	Assignment Options
Default trust configuration (SAP ID service)	<a href="#">Assign Users to Role Collections [page 2278]</a>
Custom trust configuration (for example: a tenant of SAP Cloud Identity Services)	<ul style="list-style-type: none"><li>• <a href="#">Assigning Role Collections to Users or User Groups [page 2277]</a></li><li>• <a href="#">Map Role Collections to User Groups [page 2281]</a></li><li>• <a href="#">Map Role Collections to User Attributes [page 2283]</a></li></ul>

#### Related Information

[Working with Role Collections \[page 2274\]](#)

[Trust and Federation with Identity Providers \[page 2204\]](#)

### 6.1.1.8.3.3.1 Map Role Collections to User Groups

You want to assign a role collection to a user group provided by an identity provider that has a custom trust configuration in SAP BTP. In this case, the assignment is a mapping of a user group to a role collection. Your identity provider provides the user groups using the attribute called `Groups`. Each value of the attribute is mapped to a role collection as described in this procedure.

## Prerequisites

- You've configured your custom identity provider and established trust with your subaccount.

### → Remember

The name of the trust configuration is different from SAP ID service. The name of a custom trust configuration to SAP Cloud Identity Services could be as follows:

`https://<tenant_id>.accounts.ondemand.com`

- You've configured the identity provider so that it conveys the user's group memberships in the `groups` attribute.
  - For business users, see [Map User Attributes from a Corporate Identity Provider for Business Users \[page 2208\]](#).
  - For platform users, see [Map User Attributes from a Corporate Identity Provider for Platform Users \[page 2244\]](#).
- You've created role collections.

## Context

The identity provider provides the users, who can belong to user groups. It's efficient to map user groups to role collections. The role collection as a reusable element contains the authorizations that are necessary for this user group. Mappings save time when you want to add a new user. Simply add the user to the respective user group or groups, and the user automatically gets all the authorizations that are included in the role collections.

## Procedure

To map a user group to a role collection, see [Assign User Groups to Role Collections \[page 2279\]](#).

You can also map role collections to attributes other than groups. For more information, see the related links.

## Related Information

[Trust and Federation with Identity Providers \[page 2204\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Map Role Collections to User Attributes \[page 2283\]](#)

## 6.1.1.8.3.3.2 Map Role Collections to User Attributes

Map role collections to users dynamically through the use of user attributes. When user agents present the attributes of their users, the SAP Authorization and Trust Management service can assign role collections based on the values of those attributes.

### Prerequisites

- You've configured your custom identity provider and established trust with your subaccount.
- You've configured the identity provider so that it conveys the user attributes in the assertion attributes.
  - For business users, see [Map User Attributes from a Corporate Identity Provider for Business Users \[page 2208\]](#).
  - For platform users, see [Map User Attributes from a Corporate Identity Provider for Platform Users \[page 2244\]](#).
- You've created role collections.

### Context

The identity provider hosts the users and their attributes. When a user is authenticated by the SAP Authorization and Trust Management service, a user agent presents a number of attributes of the user, such as an e-mail address. The SAP Authorization and Trust Management service uses this attribute to identify the user. You can use such attributes to assign role collections. For example, you can use the `company_region` attribute to assign role collections specific to the provinces or states your users are located. More use cases are to have different role collections for different cost centers or job functions. Mapping role collections to the attribute `Groups` is so common that we have a separate use case for it. For more information, see [Map Role Collections to User Groups](#) in the related links.

#### → Tip

Mappings save time when you add new users. Simply set the relevant attributes in the identity provider for the users, and the users automatically gets all the authorizations that are included in the role collections.

### Procedure

1. Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose ► **Security** ▶ **Role Collections** .
2. Search for and choose a role collection.
3. In the overview page for the role collection, choose the **Edit** button.
4. Under **Attribute Mapping**, select an identity provider.

5. Enter the name of the attribute and the value for the SAP Authorization and Trust Management service to use to map the role collection to users. The attributes are combined with logical OR operations. If you assign multiple attributes to the role collection, and one of them is true, the users with this attribute are assigned to this role collection.

→ Tip

Provide the **exact** name of the attribute and its value as provided by the identity provider. These values are case-sensitive.

For more information, see [Settings for Default SAML Federation Attributes of Identity Providers for Business Users \[page 2231\]](#) and [Configure the User Attributes Sent to the Application](#) in the documentation of SAP Cloud Identity Services.

6. Save your entries.

## Related Information

[Trust and Federation with Identity Providers \[page 2204\]](#)

[Map Role Collections to User Groups \[page 2281\]](#)

[Create Role Collections with Predefined Roles \[page 537\]](#)

### 6.1.1.8.4 Building Roles and Role Collections for Applications

As an administrator, you can maintain application roles and role collections which can be used in user management.

The user roles are derived from role templates that are defined in the security description (`xs-security.json`) of applications that have been registered as OAuth 2.0 clients at the User Account and Authentication service during application deployment. The application security-descriptor file also contains details of the authorization scopes that are used for application access and defines any attributes that need to be applied. The roles you create can be added to role collections.

You can perform the following tasks:

- Create and delete user roles  
User roles define authorization scopes and are based on the role templates and scopes defined in the application's security descriptor file
- Add user roles to one or more “role collections”
- Configure and manage role collections

→ Tip

Using the SAP BTP cockpit, you can assign the role collections to users directly or indirectly over attributes, such as groups.

## Related Information

[Add Roles to Role Collections on the Application Level \[page 2293\]](#)

[Mapping Role Collections in the Subaccount \[page 2281\]](#)

[Protecting Your Application \[page 501\]](#)

[Working with Role Collections \[page 2274\]](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

[Adding Authentication and Authorization \[page 499\]](#)

### 6.1.1.8.4.1 Roles and Role Collections

Usually a role collection consists of one or multiple roles. You can use the SAP BTP cockpit to add or remove roles.

#### Role

A role is an instance of a role template; you can build a role based on a role template and assign the role to a role collection. The SAP BTP cockpit helps you to display information about the selected application and any related roles in the following windows, tabs, and panes:

- Roles
- Scopes
- Attributes
- Role templates

#### Role Collection

Roles are assigned to role collections which are assigned in turn to users or user groups if an SAML 2.0 identity provider stores the users. Using the SAP BTP cockpit, you can display information about the role collections that have been maintained as well as the roles available in a role collection. Additional information includes: which templates the roles are based on, and which applications the roles apply to. Role collections enable you to group together the roles you create. The role collections you define can be assigned as follows:

- To users logged on to the SAP ID service.
- To user groups containing users logging on with SAML 2.0 assertions.

## Related Information

[Attributes \[page 2288\]](#)

## 6.1.1.8.4.2 Create Roles for Applications Using Existing Role Templates

Use the SAP BTP cockpit to create a role using an existing role template. You can refine the role by assigning attributes and add the role to role collections.

### Context

You can use existing role templates to create new roles. An SAP BTP cockpit wizard helps you to configure new roles.

#### ⓘ Note

You can only create roles from roles templates that come with attributes.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose *Security* in the navigation pane.
4. Choose *Roles*.

Here you see a complete list of all existing roles sorted by application name. It also contains the role template, role names, and role description. On the right side, you find the action buttons.

5. Choose *Create Role* in the row of the role template that you want to use.  
The *Create Role* wizard opens.
6. Enter a role name and a description, and choose *Next*.
7. Configure the attributes and choose *Next*. The attributes further refine the role. For more information, see the related link.
8. Select the available role collections for your new role and choose *Next*.
9. You can review your role configuration in the next window and complete the creation of the role by choosing *Finish*.

### Related Information

[Attributes \[page 2288\]](#)

### 6.1.1.8.4.3 Create Roles for Subscribed Applications Using Existing Role Templates

Use the SAP BTP cockpit to create a role for a subscribed application using an existing role template. You can refine the role by assigning attributes and add the role to role collections.

#### Context

You can use existing role templates to create new roles. An SAP BTP cockpit wizard helps you to configure new roles.

##### ⓘ Note

You can only create roles from roles templates that come with attributes.

#### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose in the navigation pane.
4. To see your subscribed application, choose the [Subscriptions](#) tab.
5. To get to the place where you can manage roles, choose the chevron next to your subscribed application.

Here you see a complete list of all existing roles sorted by the role template. It also contains the role names and the role description. On the right side, you find the action buttons.

6. (Optional) To add more roles, choose .
- The [Create Role](#) wizard opens.
7. Enter a role name and a description, and choose [Next](#).
  8. Configure the attributes and choose [Next](#). The attributes further refine the role. For more information, see the related link.
  9. Select the available role collections for your new role and choose [Next](#).
  10. You can review your role configuration in the next window and complete the creation of the role by choosing [Finish](#).

#### Related Information

[Configure Application Roles and Assign Roles to Users \[page 2201\]](#)  
[Attributes \[page 2288\]](#)

## 6.1.1.8.4.4 Delete Roles

Use the SAP BTP cockpit to delete a role. If this role has been added to a role collection, it's also deleted from the role collection.

### Context

You can use existing role templates to create new roles. An SAP BTP cockpit wizard helps you to configure new roles.

#### ⓘ Note

You cannot delete roles that belong to default role collections delivered by SAP BTP.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. If your role comes with a subscribed application, continue as described in [Configure Application Roles and Assign Roles to Users \[page 2201\]](#).
4. Choose *Security* in the navigation pane.
5. Choose *Roles*.

Here you see a complete list of all existing roles sorted by application name. It also contains the role template, role names, and role description. On the right side, you find the action buttons.

6. Choose the  (Delete) action button in the row of the role that you want to delete. If the role has been added to a role collection, it's also deleted from the role collection.
7. Confirm that you want to delete the role.

## 6.1.1.8.4.5 Attributes

Attributes use information that is specific to the user, for example the user's country. If the application developer in the Cloud Foundry environment of SAP BTP has created a country attribute to a role, this restricts the data a business user can see based on this attribute.

A lot of applications provide purely functional role templates which grant access for all data of a certain type within your subaccount. Roles for such role templates are generated automatically. Some other applications also provide the possibility for administrators to restrict access not only by functional authorizations, but also by instance-based authorizations. That means that users can only work with a certain subset of the data in your subaccount.

The restriction can be either based on information within the respective role, or on user-specific information provided by the identity provider. This makes instance-based authorizations specific for each customer

because the respective roles cannot be generated automatically. Instead, administrators must create them. Typical restrictions depend on information like the user's country or cost center.

Each restriction is represented by a dedicated attribute which belongs to a role template of the application.

#### Note

Application developers can define attribute references in the application security descriptor file (`xs-security.json`): For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

For each attribute, administrators have multiple options to specify the value which restricts data access:

- Static attributes  
They are stored in the role. The user is given the attribute value when the administrator assigns the role collection with this role to the user.
- Attributes from a custom identity provider  
Since an identity provider provides the business users, you can dynamically reference all attributes that come with the SAML 2.0 assertion. You define the attributes and the attribute values in the identity provider. In the Cloud Foundry environment of SAP BTP, administrators can use the attributes to refine the roles.

#### Note

If you want to reference attributes from your identity provider, you must know the exact identifier of the attribute. Go to the SAML 2.0 or OpenID Connect configuration of your identity provider and use the attributes as they are defined there.

- Unrestricted attributes

In this case, you want to express that it is not necessary to set a specific value for this attribute. The behavior is the same as if the attribute would not exist for this role.

## Related Information

[Specify Attributes in a Role \[page 2290\]](#)

[Specify Attributes in a New Role \[page 2291\]](#)

## 6.1.1.8.4.5.1 Specify Attributes in a Role

As an administrator of the Cloud Foundry environment, you can specify attributes in roles to refine authorizations of the business users. Depending on these attributes, business users with this role have restricted access to data.

### Prerequisites

You have maintained the attributes of the users in your identity provider.

#### ⓘ Note

In SAP Cloud Identity Services or any identity provider, you find the attributes in the configuration.

### Procedure

1. Open the SAP BTP cockpit.
2. Go to your subaccount.  
For more information, see [Navigate in the Cockpit \[page 2142\]](#).
3. Choose your space in [Cloud Foundry](#) [Spaces](#) or, in the case of subscriptions, see [Configure Application Roles and Assign Roles to Users \[page 2201\]](#).
4. Choose the application.
5. Choose [Security](#) [Roles](#).
6. Select a role.

The role overview pane displays the attribute name and fields where you can select the source and enter a value.

7. Choose [Edit](#).
8. To specify an attribute, choose the source of the attribute. The following sources are available:

#### Attribute Sources

Source	Value/Attribute
<a href="#">Static</a>	Enter a static value, for example <b>USA</b> to refine the role depending on the country.

#### → Remember

Follow entries with the key. Otherwise you can't save the role.

Source	Value/Attribute
<i>Identity Provider</i>	<p>Enter an attribute as defined in your identity provider. Check in your identity provider for the exact syntax of the attribute identifier.</p> <p>For SAP Cloud Identity Services, you find the attribute identifier in the settings of the attributes of your identity provider under  <i>Applications &amp; Resources</i>  <i>Applications</i>  <i>&lt;Application_Name&gt;</i>  <i>Trust</i>  <i>Attributes</i> .</p>
<i>Unrestricted</i>	<p> <b>Example</b></p> <p>To use the attribute for cost center, you must enter the value <b>cost_center</b>.</p>

9. Save your changes.

## Related Information

[Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Attributes \[page 2288\]](#)

## 6.1.1.8.4.5.2 Specify Attributes in a New Role

As an administrator, you can specify attributes in a new role to refine authorizations of business users. Depending on these attributes, business users with this role have restricted access to data.

### Prerequisites

You have maintained the attributes of the users in your identity provider.

#### Note

In SAP Cloud Identity Services or any identity provider, you find the attributes in the configuration.

## Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account (China (Shanghai) region) or subaccount. For more information, see [Navigate in the Cockpit \[page 2142\]](#).
3. Choose your space in ► *Cloud Foundry* ▶ *Spaces* ▶ or, for subscriptions, see [Configure Application Roles and Assign Roles to Users \[page 2201\]](#).
4. Choose the application.
5. Choose ► *Security* ▶ *Roles* ▶.
6. To create a new role, choose *Create*.  
A wizard guides you through the role creation process.
7. Enter a name and a description of the new role.
8. Select the role template that you want to use.
9. Choose *Next*.
10. To specify an attribute, choose the source of the attribute. The following sources are available:

### Attribute Sources

Source	Value/Attribute
<i>Static</i>	Enter a static value, for example <b>USA</b> to refine the role depending on the country/region.
<i>Identity Provider</i>	Enter an attribute as defined in your identity provider. Check in your identity provider for the exact syntax of the attribute identifier.  For SAP Cloud Identity Services, you find the attribute identifier in the settings of the attributes under ► <i>Applications &amp; Resources</i> ▶ <i>Applications</i> ▶ <Application Name> ▶ <i>Trust</i> ▶ <i>Attributes</i> ▶.
<i>Unrestricted</i>	In this case, you want to express that it isn't necessary to set a specific value for this attribute. The behavior is the same as if the attribute wouldn't exist for this role.

11. Choose *Next*.
12. Select the role collections for your new role. For more information, see the related link.
13. Choose *Next* and *Finish*.

You have now created a new role with attributes.

## Related Information

[Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#)

[Attributes \[page 2288\]](#)

[Specify Attributes in a New Role \[page 2291\]](#)

### 6.1.1.8.4.6 Add Roles to Role Collections on the Application Level

Roles are used to define the type of access granted to an application.

#### Context

A role is an instance of a role template; you can build a role based on a role template and assign the role to a role collection. Role collections are then assigned to SAML 2.0 groups or users. The role template defines the type of access permitted for an application, for example: the authorization scope, and any attributes that need to be applied. Attributes define information that comes with the respective user, for example 'cost center' or 'country' (see the related link). This information can only be resolved at run time.

#### ⓘ Note

The User Account and Authentication service automatically creates default roles for all role templates that do not include attribute references. They have the same name as the role template. You can't delete them. You can recognize them because their description contains *Default Instance*.

#### Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose *Security* in the navigation pane or go to  *Cloud Foundry*  *Spaces*  *> <your\_space>*  *> <your\_application>* .
4. Choose *Roles* in the navigation pane.

Here you see a complete list of all roles sorted by role templates. It also contains information about attributes used in this role and about the role collections the role has been added to. On the right side, you find the action buttons.

#### ⓘ Note

You cannot edit or delete predefined roles for default role collections. For this reason, the action buttons are grayed out. For more information, see the related link.

5. To directly assign a role to role collections, choose the action button  (Add) in the same row.  
A new window shows all role collections that are available in your global account or subaccount.
6. Select the role collections to which you want to add your role.
7. Choose *Add*.

The number in the role collections column counts up. You have now assigned this role to the role collections you selected earlier.

## Related Information

[Attributes \[page 2288\]](#)

[Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

### 6.1.1.8.5 Managing Secrets of the SAP Authorization and Trust Management Service

The SAP Authorization and Trust Management service maintains a number of secrets to ensure secure operation of the service. Your organization can have policies that require you change secrets or you may need to respond to the loss of a secret.

The SAP Authorization and Trust Management service uses the following secrets in its operation:

- [Service Instance Secrets \[page 2294\]](#)
- [Rotate Binding Secrets \[page 2297\]](#)
- [Migrate from Instance Secrets to Binding Secrets \[page 2299\]](#)
- [Rotating Instance Secrets \[page 2300\]](#)
- [Binding Parameters of SAP Authorization and Trust Management Service \[page 2301\]](#)
- [Rotate Signing Keys of Access Tokens \[page 2308\]](#)
- [Rotate Signing Keys of SAML Token \[page 2312\]](#)

#### 6.1.1.8.5.1 Service Instance Secrets

When an application consumes a service instance of the SAP Authorization and Trust Management service (XSUAA), the application identifies itself to the service instance with a client ID and a secret. The client ID and secret are the credentials with which an application authenticates itself to the service instance.

The system creates these credentials either when you bind the application to the service instance or when you create a service key for the service instance.

The service instance can use multiple secrets in the application plan.

- The instance secret is the default secret type. The secret is the same for all bindings of the service instance. The secret remains valid as long as the instance exists.

#### ⓘ Note

Instance secrets are supported only in the Cloud Foundry environments with just one organization. For multiple organizations or other environments, binding secrets are the default secret type. For more information, see [Rotating Secrets](#).

- Binding secrets must be enabled in the application security descriptor (`xs-security.json`) when you create the service instance. When you bind an application to a service instance or create a service key, you can pass a `parameters.json` to use a binding secret. The secret remains valid as long as the binding or the service key exists.

#### ⓘ Note

The `apiaccess` plan only uses binding secrets. However, some old instances of the `apiaccess` plan still use the instance secret.

- X.509 secrets must be enabled in the application security descriptor (`xs-security.json`) when you create the service instance. When you bind an application to a service instance or create a service key, you can pass a `parameters.json` to use an X.509 secret. SAP Authorization and Trust Management service can generate an X.509 certificate for you or you can provide your own. The X.509 secret remains valid as long as the certificate itself is valid.

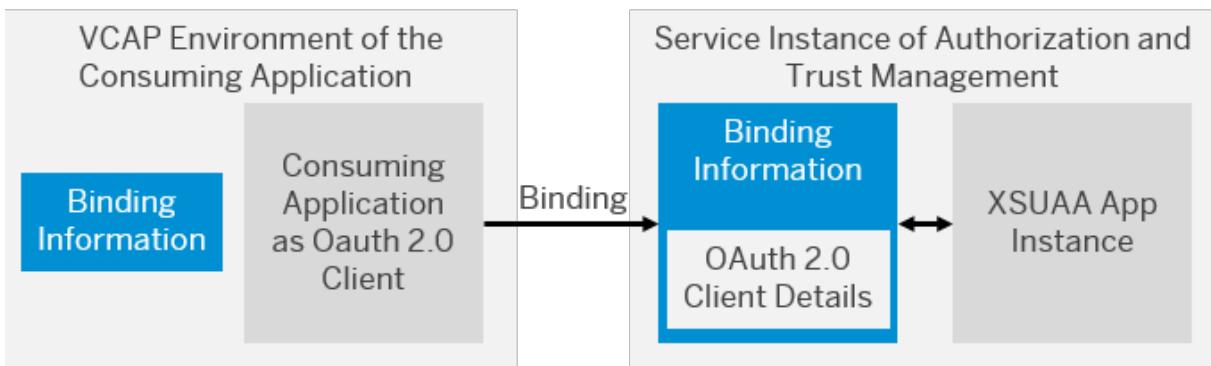
#### ⓘ Note

The service doesn't check for certificate revocation. To stop the service from accepting a certificate that is still valid, delete the relevant bindings or service keys. As soon as the binding is deleted, the service stops accepting the certificate.

For bindings with self-managed certificates and the `certificate-pinning` parameter set to `false`, you can rotate the secrets without deleting bindings. Just use a new certificate with the same subject and issuer distinguished name (DN). The service saves the new validity date of the new certificate.

For more information, see [Parameters for Self-Managed X.509 Certificates \[page 2304\]](#).

The following figure illustrates the XSUAA app and the binding information about the OAuth 2.0 client as part of a service instance of the SAP Authorization and Trust Management service. A consuming application, functioning as an OAuth 2.0 client is bound to the SAP Authorization and Trust Management service instance. The secret is part of the environment of the consuming application and the information about the OAuth 2.0 client saved with the XSUAA app. Alternatively, this information is saved as part of a service key.



Binding Between an SAP Authorization and Trust Management Service Instance and a Consuming Application

The `credential-types` parameter of the OAuth client configuration in the application security descriptor (`xs-security.json`) determines which types of secrets that bindings support.

In the following example, the service instance creates a binding secret for all new bindings. It allows the creation of X.509 secrets and still accepts instance secrets.

#### Example

##### Sample Code

```
"oauth2-configuration": {
    "credential-types": [ "binding-secret", "x509", "instance-secret" ]
}
```

In the following example, the service instance creates a binding secret for all new bindings, but doesn't accept the instance secret or the use of X.509 secrets.

#### Example

##### Sample Code

```
"oauth2-configuration": {
    "credential-types": [ "binding-secret" ]
}
```

## Related Information

[Rotate Binding Secrets \[page 2297\]](#)

[Rotating Instance Secrets \[page 2300\]](#)

[Migrate from Instance Secrets to Binding Secrets \[page 2299\]](#)

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

## 6.1.1.8.5.2 Rotate Binding Secrets

Service instances of the SAP Authorization and Trust Management service use different binding secrets for each binding. To rotate binding secrets, unbind and rebinding any consuming applications.

### Prerequisites

- You've enabled support for binding secrets.  
For more information, see [Migrate from Instance Secrets to Binding Secrets \[page 2299\]](#).
- If you have to replace any service keys, create the new service keys before proceeding.  
For more information, see [Creating Service Keys \[page 255\]](#).

### Procedure

1. Unbind the application that consumes your service instance.

Use the following syntax:

```
cf unbind-service <APP_NAME> <SERVICE_INSTANCE>
```

```
cf unbind-service my-app my-xsuaa-service
```

2. Rebind the application that consumes your service instance.

Use the following syntax:

```
cf bind-service <APP_NAME> <SERVICE_INSTANCE> [-c <PARAMETERS_AS_JSON>]
```

```
cf bind-service my-app my-xsuaa-service
```

### Results

When you rebinding the application, the system creates a new binding secret.

### Next Steps

Delete any old service keys that you don't need anymore.

## Related Information

[Service Instance Secrets \[page 2294\]](#)

### 6.1.1.8.5.3 Rotate X.509 Secrets

When your private key is exposed or your certificates are about to expire, it's time to rotate secrets. Rotating secrets can be as easy as unbinding and rebinding your application or service key.

#### Prerequisites

- If you provide your own certificates, you've prepared new certificates.
- If you need to replace any service keys, create the new service keys before proceeding.  
For more information, see [Creating Service Keys \[page 255\]](#).

#### Context

##### ⓘ Note

The service doesn't check for certificate revocation. To stop the service from accepting a certificate that is still valid, delete the relevant bindings or service keys. As soon as the binding is deleted, the service stops accepting the certificate.

For bindings with self-managed certificates and the `certificate-pinning` parameter set to `false`, you can rotate the secrets without deleting bindings. Just use a new certificate with the same subject and issuer distinguished name (DN). The service saves the new validity date of the new certificate.

For more information, see [Parameters for Self-Managed X.509 Certificates \[page 2304\]](#).

#### Procedure

1. Unbind the application that consumes your service instance.

Use the following syntax:

```
cf unbind-service <APP_NAME> <SERVICE_INSTANCE>
```

```
cf unbind-service my-app my-xsuaa-service
```

2. Rebind the application that consumes your service instance.

Use the following syntax:

```
cf bind-service <APP_NAME> <SERVICE_INSTANCE> [-c <PARAMETERS_AS_JSON>]
```

```
cf bind-service my-app my-xsuaa-service -c parameters.json
```

If you provide your own certificates, include them in a `parameters.json` file.

## Related Information

[Binding Parameters of SAP Authorization and Trust Management Service \[page 2301\]](#)

[Binding Service Instances to Applications \[page 253\]](#)

[Service Instance Secrets \[page 2294\]](#)

### 6.1.1.8.5.4 Migrate from Instance Secrets to Binding Secrets

To simplify the management of secrets for service instances of the SAP Authorization and Trust Management service, we recommend that you configure service instances to use binding secrets.

#### Context

By default, service instances of the SAP Authorization and Trust Management service use the instance secret for all bindings of the service instance. In the application security descriptor (`xs-security.json`), enable binding secrets for service instances. All bindings have their own secret. You can enable both at once for the following plans:

- Application plan

The API access plan only uses binding secrets.

#### Procedure

1. Modify the application security descriptor (`xs-security.json` service use the instance) to support both instance secrets and binding secrets.

Edit the OAuth client configuration of the `xs-security.json` as follows:

##### Sample Code

```
"oauth2-configuration": {  
    "credential-types": [ "binding-secret", "instance-secret" ]  
}
```

2. Update the service instance with the new application security descriptor.
3. Unbind and rebind any consuming applications.

With each new binding, the system creates a new binding secret.

4. Replace any service keys with new service keys.

At this point, none of the applications consuming your service instance need the instance secret anymore.

5. Modify the application security descriptor (`xs-security.json`) to disable support for instance secrets.

Edit the OAuth client configuration of the `xs-security.json` as follows:

#### ↔ Sample Code

```
"oauth2-configuration": {  
    "credential-types": [ "binding-secret" ]  
}
```

6. Update the service instance with the new application security descriptor.

## Related Information

[Update a Service Instance \[page 530\]](#)

[Creating Service Keys \[page 255\]](#)

[Service Instance Secrets \[page 2294\]](#)

## 6.1.1.8.5.5 Rotating Instance Secrets

When configured for instance secrets, a service instance of the SAP Authorization and Trust Management service uses the same instance secret for all bindings. You can't really rotate instance secrets, but must rotate the applications and service instance together.

#### → Recommendation

Migrate to using binding secrets instead of instance secrets.

Using instance secrets forces you to redeploy the entire application under a new `xsappname` and repeat the assignment of any role collections.

For more information, see [Migrate from Instance Secrets to Binding Secrets \[page 2299\]](#).

## Related Information

[Service Instance Secrets \[page 2294\]](#)

## 6.1.1.8.5.6 Binding Parameters of SAP Authorization and Trust Management Service

When binding applications or creating service keys for services instances of the SAP Authorization and Trust Management service (XSUAA), provide configuration parameters in JSON format.

Set parameters according to the following use cases:

- For instance or binding secrets, you want to suppress the creation of client secrets.
- For X.509 secrets, set parameters according to the following scenarios:
  - The SAP Authorization and Trust Management service generates certificates for you.
  - You already have your own public key infrastructure (PKI), with certificates issued from one of the supported certificate authorities (CA). See the links in the related information.

### → Remember

The types of secrets that a service instance can accept in a binding or service key is configured in the application security descriptor (`xs-security.json`).

For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

## Related Information

[Parameters for Suppressing Client Secrets \[page 2301\]](#)

[Parameters for X.509 Certificates Managed by SAP Authorization and Trust Management Service \[page 2303\]](#)

[Parameters for Self-Managed X.509 Certificates \[page 2304\]](#)

[Trusted Authorities for X.509 Certificates \[page 2307\]](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

## 6.1.1.8.5.6.1 Parameters for Suppressing Client Secrets

When binding or creating a service key for an `xsuaa` service instance, you can pass some parameters in JSON format or in a JSON file (see `cf bind-service` and `cf create-service-key` in the related links). The "hide-secret" element enables you to suppress the client secret when binding or creating a service key. It's useful if some applications only want to bind the `xsuaa` service for authorization purposes.

## Prerequisites

- In the application security descriptor file (`xs-security.json`), you've included the "credential-type" key with "instance-secret" and/or "binding-secret" when you created or updated the service instance. The "hide-secret" element is only taken into account if the credential type is "instance-secret" or "binding-secret".

### Note

If the "credential-type" element isn't defined in the binding descriptor, it defaults to the first value of the array passed as "credential-types" in xs-security.json for the service instance or for the first value of the default list.

For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

## Parameters for Suppressing Client Secrets

The `cf create-binding` or `cf create-service-key` commands expect a valid `parameters.json` file containing the service-specific configuration parameters in the following format.

### Sample Code

```
{  
  "credential-type": "instance-secret",  
  "hide-secret": "true"  
}
```

### Sample Code

```
{  
  "credential-type": "binding-secret",  
  "hide-secret": "true"  
}
```

The `hide-secret` element defines whether the client secret is omitted.

Element for Suppressing Client Secrets

Element	Value	Description
hide-secret	<code>true</code>	The client secret is omitted in the binding information.
	<code>false</code> (default)	The client secret is included in the binding information.

## Related Information

[Cloud Foundry CLI Reference](#) 

## 6.1.1.8.5.6.2 Parameters for X.509 Certificates Managed by SAP Authorization and Trust Management Service

Use the parameters to have the service generate X.509 certificates for you.

### Prerequisites

You've enabled the service instance to use X.509 secrets.

For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

### Parameters for Certificates Managed by the Service

The `create-binding` or `create-service-key` commands expect a `parameters.json` in the following format.

```
{  
  "credential-type": "x509",  
  "x509": {  
    "key-length": 2048,  
    "validity": 8,  
    "validity-type": "DAYS"  
  }  
}
```

The following table describes the supported parameters.

Parameters for Service-Managed Certificates

Parameter	Description
credential-type	Only required to use X.509 when X.509 isn't specified as default in the application security descriptor ( <code>xs-security.json</code> ).  For more information, see <a href="#">Application Security Descriptor Configuration Syntax [page 545]</a> .
x509	In this parameter, you define the certificate options of key length, validity period, and unit of length. This parameter is optional.
key-length	Specifies the byte length of the generated private key. The default length is 2048 bytes. You can also choose 4096 bytes or 8192 bytes.

Parameter	Description
validity	<p>Specifies the number of time units for validity-type. The default value is 7.</p> <p>Together with the validity-type the range of validity runs from 1 DAYS to 30 DAYS.</p>
validity-type	<p>Specifies the time unit for validity. Supported value is DAYS. The default value is DAYS.</p>

## Related Information

[Service Instance Secrets \[page 2294\]](#)

### 6.1.1.8.5.6.3 Parameters for Self-Managed X.509 Certificates

Use these parameters to provide your own certificates for a binding or service key to service instances of the SAP Authorization and Trust Management service (XSUAA).

## Prerequisites

- You've enabled the service instance to use X.509 secrets.  
For more information, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).
- To use your own public key infrastructure (PKI), the certificates must be issued from one of the following certificate authorities (CA).  
For more information, see [Trusted Authorities for X.509 Certificates \[page 2307\]](#).

## Parameters for Self-Managed Certificates

The create binding or create service key command expects a `parameters.json` in the following format.

```
{
  "credential-type": "x509",
  "x509": {
    "certificate": "-----BEGIN CERTIFICATE-----...-----END CERTIFICATE-----",
    "ensure-uniqueness": false,
    "certificate-pinning": true,
    "hide-certificate": true
  }
}
```

## Parameters for Self-Managed Certificates

Parameter	Description
credential-type	<p>Only required to use X.509 when X.509 isn't specified as default in the application security descriptor (<code>xs-security.json</code>).</p> <p>For more information, see <a href="#">Application Security Descriptor Configuration Syntax [page 545]</a>.</p>
x509	In this parameter, you include the certificate and the options of uniqueness and pinning. This parameter is required.
certificate	Provide the certificate string in privacy-enhanced mail (PEM) format. This parameter is required.
	<p><b>⚠ Restriction</b></p> <p>You can only use each certificate once per service instance. The uniqueness of the certificate among all bindings and keys of a service instance is dependent on the value of <code>certificate-pinning</code>:</p> <ul style="list-style-type: none"><li>• <code>true</code> The certificate hash must be unique.</li><li>• <code>false</code> The combination of subject and issuer DN must be unique.</li></ul>
ensure-uniqueness	Ensures that the certificate is unique among all following clone instances. The default value is <code>false</code> .
	<p><b>ⓘ Note</b></p> <p>This parameter only applies to the broker plan of the service.</p>

Parameter	Description
certificate-pinning	<p>Set to <code>false</code> for applications that are rarely updated or deployed. The incoming certificate's subject and issuer DN is compared to the subject and issuer of the stored certificate. If the values match and the certificate was issued on or after the stored issuer date, the authentication is accepted. Afterwards the incoming certificate's issuer date is stored for future authentication attempts.</p> <p><b>Exception:</b> Matching certificates with a <code>valid_from</code> parameter that is less than 10 minutes old are always accepted.</p> <p>Set to <code>true</code> for blue and green deployments, where you deploy a new application or version of an application regularly and rotate the certificate in parallel. For this use case, you need a new certificate to use in the binding of the new application.</p> <p>The default value is <code>true</code>.</p>
hide-certificate	<p>If this parameter is set to <code>true</code> (default is <code>false</code>), the X.509 certificate is not returned in the binding or service key. This is useful to reduce the size of the environments when the Cloud Foundry environment limit of 130k has been reached. This parameter is optional.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>This parameter works only with self-managed certificates as otherwise, the public key would not be known to the user.</p> </div>

## Related Information

[Service Instance Secrets \[page 2294\]](#)

## 6.1.1.8.5.6.4 Trusted Authorities for X.509 Certificates

Service instances of the SAP Authorization and Trust Management service (XSUAA) trust the following certificate authorities (CA). To use your own public key infrastructure (PKI) for bindings, the certificates must be issued from one of these CAs.

Supported CAs for X.509 Certificates

Subject	Issuer	Validity
C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert Global Root G2	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert Global Root G2	Jan 15 12:00:00 2038 GMT
C=DE, L=Walldorf, O=SAP AG, CN=SAP Global Root CA	C=DE, L=Walldorf, O=SAP AG, CN=SAP Global Root CA	Apr 26 15:46:27 2032 GMT
C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert Global Root CA	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert Global Root CA	Nov 10 00:00:00 2031 GMT
C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert SHA2 Secure Server CA	C=US, O=DigiCert Inc, OU=www.digicert.com, CN=DigiCert Global Root CA	Sep 22 23:59:59 2030 GMT
C=US, O=Entrust, Inc., OU=www.entrust.net/legal-terms, CN=Entrust Root Certification Authority - G2	C=US, O=Entrust, Inc., OU=www.entrust.net/legal-terms, CN=Entrust Root Certification Authority - G2	Dec 07 17:55:54 2030 GMT
C=IE, O=Baltimore, OU=CyberTrust, CN=Baltimore CyberTrust Root	C=IE, O=Baltimore, OU=CyberTrust, CN=Baltimore CyberTrust Root	May 12 23:59:00 2025 GMT

### Related Information

[Parameters for Self-Managed X.509 Certificates \[page 2304\]](#)

[Service Instance Secrets \[page 2294\]](#)

## 6.1.1.8.5.7 Rotate Signing Keys of Access Tokens

Components of the Cloud Foundry environment use the digital signature of the access tokens to verify the validity of access tokens. Periodically rotate the signing keys of access tokens.

### Context

### Related Information

[Rotate Signing Keys of Access Tokens with the SAP BTP CLI \[page 2308\]](#)

[Rotate Signing Keys of Access Tokens with the Security Setting API \[page 2310\]](#)

### 6.1.1.8.5.7.1 Rotate Signing Keys of Access Tokens with the SAP BTP CLI

The SAP BTP CLI enables you to rotate the signing keys of access tokens.

### Prerequisites

Plan for waiting time in hours between the steps of this procedure.

After activating a new key, wait for access tokens signed by the old key to expire before deleting the old key. The default lifetime of access tokens is 12 hours.

To support this delay in rotation, the service enforces a minimum 1-hour delay between rotation and the activation.

#### Note

In emergency cases or in automated test setups, you can bypass the delay in activation and deletion by using the `--force` option. Forcing changes can also cause service interruption for access tokens that haven't expired yet.

### Context

To rotate the signing key of access tokens, use the SAP BTP command-line interface (btp CLI). You can rotate signing keys for the global account, directory, or subaccount.

For more information, see [Managing Signing Keys for Access Tokens \[page 2368\]](#).

## Procedure

1. Log on to the SAP BTP CLI.  
For more information, see [Log in \[page 2337\]](#).
2. Check that there's space for a new signing key.

You can store two signing keys per subaccount.

Use the following command:

```
btp list security/token-key
```

The service lists any keys.

Key ID	Enabled	Created	Fips
Compliant			
default-jwt-key-491aae9c42		Fri Feb 02 14:48:51 UTC 2024	yes
default-jwt-key-ca81ebbb43	yes	Mon Feb 05 14:42:26 UTC 2024	yes
2 entries			
OK			

Delete the inactive key if you already have two. In the previous example, delete default-jwt-key-491aae9c42. For more information on how to delete keys, see step 6.

### ⚠ Caution

If the keyID is key-id-0 or key-id-1, these keys are legacy signing keys. When you enable the first signing key after a legacy signing key, the legacy signing key is immediately invalid. Clients with access tokens issued within the lifetime of an access token can't use their access tokens anymore. Affected clients must reauthenticate to get a new token.

In addition, the issuer of the key changes as well. Update the configuration of systems depending on these keys, such as SAP HANA systems and cloud connector connections. For more information, see:

- [Managed Certificate Collections](#) in the *SAP HANA Cloud, SAP HANA Database Security Guide*
- [Configure a CA Certificate](#) in the *SAP BTP Connectivity* documentation

Plan for possible service interruption when you add your first signing key.

3. Add a new signing key for the access token.

Use the following command:

```
btp create security/token-key  
Created default signing key
```

A new signing key is generated.

4. Enable the new signing key.

Use the following command:

```
btp enable security/token-key --key <Key_ID>
```

For example:

```
btp enable security/token-key --key default-jwt-key-ca81ebbb43  
Enabled signing key 'default-jwt-key-ca81ebbb43'
```

New access tokens are now signed with the new signing key.

5. Wait for access tokens signed by the old key to expire.

As long as the old signing key exists, the system still accepts digital signatures signed by that key. Once you've waited out the lifetime of any access tokens signed by the old key, you can delete the old key.

6. Delete the old signing key for the access token.

Use the following command:

```
btp delete security/token-key --key <Key_ID>
```

For example:

```
btp delete security/token-key --key default-jwt-key-491aae9c42  
Do you really want to delete signing key 'default-jwt-key-491aae9c42'? [no]>  
Yes  
Deleted signing key 'default-jwt-key-491aae9c42'
```

## 6.1.1.8.5.7.2 Rotate Signing Keys of Access Tokens with the Security Setting API

You can use the Security Setting API of the SAP Authorization and Trust Management service to rotate the signing keys of access tokens .

### Prerequisites

- You've enabled API access to the service.  
For more information, see [Get Access to the APIs \[page 2417\]](#).
- Plan for waiting time in hours between the steps of this procedure.  
After activating a new key, wait for access tokens signed by the old key to expire before deleting the old key. The default lifetime of access tokens is 12 hours.  
To support this delay in rotation, the service enforces a minimum 1-hour delay between rotation and the activation (UPDATE) or deletion (DELETE) of signing keys.

#### ⓘ Note

In emergency cases or in automated test setups, you can bypass the delay in activation and deletion by using the FORCE\_UPDATE and FORCE\_DELETE change modes. Forcing changes can also cause service interruption for access tokens that haven't expired yet.

## Context

## Procedure

1. Check that there's space for a new signing key.

You can store two signing keys per subaccount. Get the settings of your subaccount to see how many keys are stored there.

Call the [GET](#) method of the Security Settings API at the following endpoint:

```
https://api.authentication.<region>.hana.ondemand.com/sap/rest/authorization/v2/securitySettings
```

For more information, check the security settings API on the [SAP Business Accelerator Hub](#).

The API returns a JSON with the security settings. Examine the key IDs in the token policy settings.

```
{  
  ...  
  "tokenPolicySettings": {  
    ...  
    "activeKeyId": "jwt-sig-2022-12-01",  
    "keyIds": [  
      "jwt-sig-2022-09-10",  
      "jwt-sig-2022-12-01"  
    ]  
  },  
  ...  
}
```

Delete the inactive key if you already have two. In the previous example, delete `jwt-sig-2022-09-10`. For more information on how to delete keys, see step 5.

### ⚠ Caution

If the `keyID` is `key-id-0` or `key-id-1`, these keys are legacy signing keys. When you enable the first signing key after a legacy signing key, the legacy signing key is immediately invalid. Clients with access tokens issued within the lifetime of an access token, can't use their access tokens anymore. Affected clients must reauthenticate to get a new token.

In addition, the issuer of the key changes as well. Update the configuration of systems depending on these keys, such as SAP HANA systems and cloud connector connections. For more information, see:

- [Managed Certificate Collections](#) in the *SAP HANA Cloud, SAP HANA Database Security Guide*
- [Configure a CA Certificate](#) in the *SAP BTP Connectivity* documentation

Plan for possible service interruption when you add your first signing key.

2. Add a new signing key for the access token.

Call the [PATCH](#) method of the same endpoint.

In the body, include a key ID and set `changeMode` to `ADD` in the token policy settings.

#### ↔ Sample Code

```
{  
    "tokenPolicySettings": {  
        "keyId": "jwt-sig-2023-01-30",  
        "changeMode": "ADD"  
    }  
}
```

A new signing key is generated.

3. Enable the new signing key.

Call the [PATCH](#) method of the same endpoint, but set `changeMode` to [UPDATE](#).

```
{  
    "tokenPolicySettings": {  
        "keyId": "jwt-sig-2023-01-30",  
        "changeMode": "UPDATE"  
    }  
}
```

New access tokens are now signed with the new signing key.

4. Wait for access tokens signed by the old key to expire.

As long as the old signing key exists, the system still accepts digital signatures signed by that key. Once you've waited out the lifetime of any access tokens signed by the old key, you can delete the old key.

5. Delete the old signing key for the access token.

Call the [PATCH](#) method of the same endpoint, but set `keyID` to the old key ID and `changeMode` to [DELETE](#).

#### ↔ Sample Code

```
{  
    "tokenPolicySettings": {  
        "keyId": "jwt-sig-2022-12-01",  
        "changeMode": "DELETE"  
    }  
}
```

## Related Information

[Managing Secrets of the SAP Authorization and Trust Management Service \[page 2294\]](#)

## 6.1.1.8.5.8 Rotate Signing Keys of SAML Token

The SAP Authorization and Trust Management service uses an X.509 certificate to sign SAML 2.0 protocol messages between itself, a SAML service provider, and an SAML identity provider. To rotate the signing keys for SAML tokens, use the Security Settings API of the SAP Authorization and Trust Management service and then establish trust by exchanging metadata between the service provider and identity provider.

## Prerequisites

- You've enabled API access to the SAP Authorization and Trust Management service.  
For more information, see [Get Access to the APIs \[page 2417\]](#).
- Optionally, to use your own signing key for SAML tokens, you've obtained an X.509 certificate for the SAP Authorization and Trust Management service.
- You've checked that you've space for an additional signing key.  
You can store two signing keys per subaccount. Get the settings of your subaccount to see how many signing keys you've stored there. Delete an inactive signing key if you already have two.

## Context

The signed SAML tokens are used to validate and authenticate the protocol messages between the SAP Authorization and Trust Management service instance and the identity provider.

## Procedure

1. Add a new signing key for SAML tokens.

Call the [PATCH](#) method of the Security Settings API at the following endpoint:

```
https://api.authentication.<region>.hana.ondemand.com/sap/rest/authorization/v2/securitySettings
```

In the body, include a key ID and set `changeMode` to [ADD](#) in the SAML configuration settings.

### ↔ Sample Code

```
{  
    "samlConfigSettings": {  
        "keyId": "my-new-key",  
        "changeMode": "ADD"  
    }  
}
```

A new signing key is generated.

Optionally, provide your own signing key with the `key` parameter. The signing key is your private key of the certificate key pair. Provide the certificate.

### → Recommendation

We recommend providing an empty passphrase. We believe that there's no reason to view the private key.

```
{  
    "samlConfigSettings": {  
        "keyId": "my-new-key",  
        "passphrase": ""  
    }  
}
```

```
        "changeMode": "ADD",
        "key": [
            {
                "key": "<key_data>",
                "passphrase": "",
                "certificate": "<certificate_data>"
            }
        ]
    }
```

2. Enable the new signing key.

Call the [PATCH](#) method of the same endpoint, but set `changeMode` to **UPDATE**.

```
{
    "samlConfigSettings": [
        {
            "keyId": "my-new-key",
            "changeMode": "UPDATE"
        }
    ]
}
```

New SAML tokens are now signed with the new signing key.

3. Establish trust with the identity providers you consume.

Upload the new metadata information to your identity providers.

For more information, see the related documentation:

- [Manually Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2224\]](#)
- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 2228\]](#)

### ⚠ Caution

Until you upload the metadata to your identity provider, the identity provider rejects the signed SAML tokens.

4. Delete the old signing key for SAML tokens.

Call the [PATCH](#) method of the same endpoint, but set `keyID` to the old key ID and `changeMode` to **DELETE**.

### ↔ Sample Code

```
{
    "samlConfigSettings": [
        {
            "keyId": "my-old-key",
            "changeMode": "DELETE"
        }
    ]
}
```

## Related Information

[Managing Secrets of the SAP Authorization and Trust Management Service \[page 2294\]](#)

## 6.1.1.8.6 Configure Trusted Domains for Multi-environment Subaccounts

By default, applications of the subaccount, including login pages of the SAP Authorization and Trust Management service (XSUAA), can't be framed by other applications in different domains for security reasons.

### Prerequisites

- Recommendation  
Review the security implications of using inline frames (IFrames) with the SAP Authorization and Trust Management service.  
For more information, see [Implications of Using IFRAMES \[page 3074\]](#).
- Check that your application supports the trusted domains of multi-environment subaccounts.  
The documentation of your application indicates whether you need to make this configuration or not.  
The login pages of the SAP Authorization and Trust Management service (XSUAA) supports this feature.

### Context

To prevent clickjacking or overlay attacks, web browsers follow a same origin policy. The same origin policy means you can only embedded content in your web page if the content shares the same protocol, host, and port as your web page. If the origins don't match, you must maintain a list of origins that allows for exceptions to the same origin policy. The same origin policy also applies to cookies. Cookies enable single sign-on scenarios. If the cookies have a different origin, the browser rejects them like any other content.

#### → Recommendation

Instead of configuring lists of trusted domains, ensure that your application runs under the same domain as the framing application.

For more information about the SAP Custom Domain service, see the documentation of the [SAP Custom Domain Service](#).

If your custom identity provider runs under a different domain as the framing application, you must configure the identity provider to trust the domain of the framing application, too.

### Procedure

- Go to your subaccount (see [Navigate in the Cockpit \[page 2142\]](#)) and choose ► **Security** ► **Settings** ▾.
- Choose **+** (**Add**).

3. Enter the host name for the trusted domain.

For example: `https://store.example.com`

Wildcards are allowed for the subdomain.

For example: `https://*.example.com`

 **Note**

All the trusted domains together, including their space separators, can't exceed 2048 characters.

4. Save your entries.

## Next Steps

Ensure that other components framed by the host application also trust the framing domain. Typical components include the application router and the identity provider.

## Related Information

[Application Router \[page 409\]](#)

[Configure Trust Domains \(in SAP Cloud Identity Services - Identity Authentication Documentation\)](#)

### 6.1.1.8.7 Configure Token Policy for SAP Authorization and Trust Management Service

Set the token policy in the SAP BTP cockpit for SAP Authorization and Trust Management service by configuring the validity of the OpenID Connect (OIDC) tokens the service issues.

## Prerequisites

→ Recommendation

Review the security implications of token policies for SAP Authorization and Trust Management service.

For more information, see [Setting Token Policy \[page 3076\]](#).

## Context

On request, SAP Authorization and Trust Management service issues access and refresh tokens.

With a valid access token, you can access a protected resource. Once an access token expires, you can get new access tokens with a refresh token. Once the refresh token expires, you must reauthenticate and request new access and refresh tokens.

### → Recommendation

Relaxing the token policy means that users reauthenticate less. However, increasing the token validity also means that if a malicious user manages to steal a token, that malicious user has access until the token expires. Keep token validity as short as possible, but not less than 30 minutes.

To change the token validity, access the security settings. The change applies to all service instances in the subaccount that haven't set a specific value in the application security descriptor (`xs-security.json`).

## Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account and subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose  **Security** .
4. Choose the *Token Validity* tab.

You see the setting of the current access token validity and of the current refresh token validity.

5. Use the slider to set the validity for each token. The tooltip shows the selected value in seconds.

### ❖ Example

- Set the access token validity to 1800 seconds
- Set the refresh token validity to 43200 seconds.

6. To save the value, choose *Set Token Validity*.

### ⓘ Note

It is also possible to change the token validity in the Security Settings API.

For more information, see [Security Settings API](#)  on [SAP Business Accelerator Hub](#) .

## Related Information

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

## 6.1.2 Account Administration Using Joule (Beta)

Learn about the administrative tasks that SAP's generative AI copilot, Joule, can perform for you in the SAP BTP cockpit.

### ⓘ Note

This offering is a beta feature. Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. For more information, see [Important Disclaimers and Legal Information](#).

Joule makes it easier than ever to create subaccounts and manage account users in the cockpit. With Joule integrated into the cockpit, you can accomplish your goals faster.

- Ask Joule to perform an administrative action for you from a set of pre-defined capabilities.
- Ask Joule a question about SAP BTP services or the cockpit. Joule references SAP BTP cockpit documentation and provides a summary of what it finds, as well as links to relevant help content.

#### [Access Joule \(Beta\) \[page 2318\]](#)

Launch Joule directly from SAP BTP cockpit.

#### [Supported Administrative Tasks \(Beta\) \[page 2319\]](#)

Learn about the administrative tasks that Joule can perform for you in the SAP BTP cockpit.

#### [Conversations with Joule \(Beta\) \[page 2320\]](#)

Let Joule help you find answers to questions about managing your accounts in SAP BTP cockpit.

#### [Troubleshooting Joule in SAP BTP cockpit \[page 2321\]](#)

When you work with Joule, you could have questions about the things you see in the chat conversation.

A list of common questions and answers are provided here.

## Related Information

[What is Joule?](#)

[Using Joule](#)

### 6.1.2.1 Access Joule (Beta)

Launch Joule directly from SAP BTP cockpit.

### ⓘ Note

This offering is a beta feature. Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. For more information, see [Important Disclaimers and Legal Information](#).

Joule is accessed directly from the cockpit when you click the diamond icon  in the cockpit tool header. When the chat window opens, you can ask Joule to perform one of the supported capabilities or you can ask Joule a question about SAP BTP.

## Authentication

Joule only supports SAP-managed IAS tenants for authentication at this time.

## Authorization

Because you're already logged into the SAP BTP cockpit, Joule recognizes you and can perform administrative actions on your behalf. It's important to know that Joule is bound to your cockpit authorization. In other words, Joule can only accomplish tasks for you that you're authorized to perform.

## Related Information

[Supported Browsers](#)

[Supported Administrative Tasks \(Beta\) \[page 2319\]](#)

[Conversations with Joule \(Beta\) \[page 2320\]](#)

[Using Joule](#)

### 6.1.2.2 Supported Administrative Tasks (Beta)

Learn about the administrative tasks that Joule can perform for you in the SAP BTP cockpit.

#### ⓘ Note

This offering is a beta feature. Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. For more information, see [Important Disclaimers and Legal Information](#).

Joule doesn't support administrative actions in Neo environments.

You can ask Joule to perform certain administrative tasks for you in the cockpit. If Joule recognizes the task as a supported capability, then it walks you through the steps to complete the task.

You can ask Joule to do these things in the cockpit:

- Create a subaccount.

#### ⓘ Note

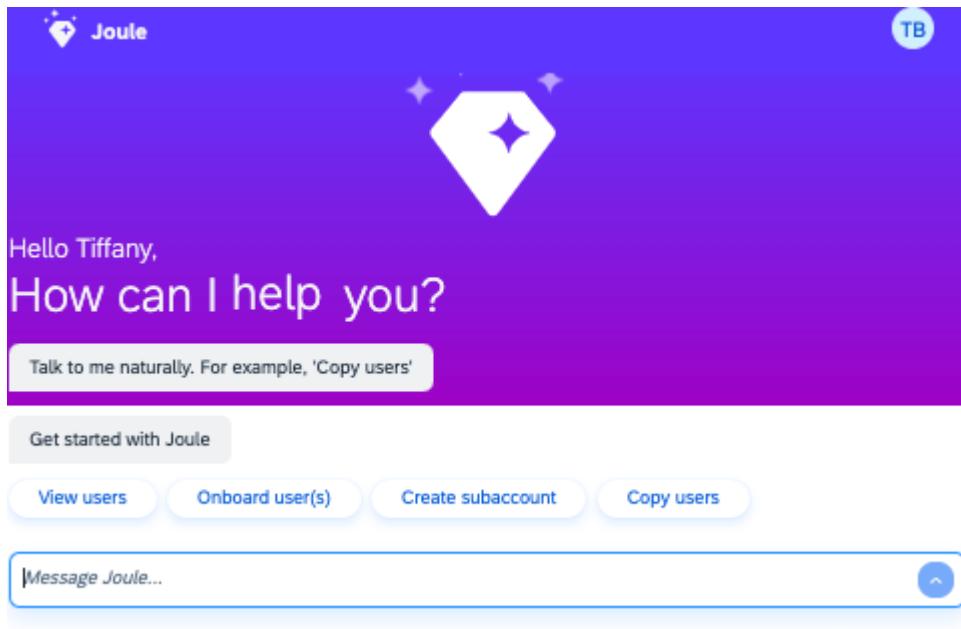
Subaccounts are created with a beta label to identify that they're part of the beta program. These subaccounts aren't available in global accounts that are used in production.

- Enable Cloud Foundry for a subaccount. This option is available as an action button in the chat window after you create a new subaccount.
- Add a new user to a subaccount. You can assign a role collection once the user is created.
- Copy users from a global account to a subaccount. By default, users are assigned the **Subaccount Viewer** role.

- Copy users from one subaccount to another subaccount. Users are copied over with the same role permissions that they were assigned in the source subaccount, as long as that role already exists in the destination subaccount.

There are times when it's helpful to see a summary of your global account and subaccount user details. You can also ask Joule to show you these things in the cockpit:

- View global account users. You can view all users or filter by role.
- View subaccount users. You can view all users or filter by role.
- View subaccount details. Joule displays information about the account. For example, region, Cloud Foundry status, subdomain, and more.



### 6.1.2.3 Conversations with Joule (Beta)

Let Joule help you find answers to questions about managing your accounts in SAP BTP cockpit.

#### *ⓘ* Note

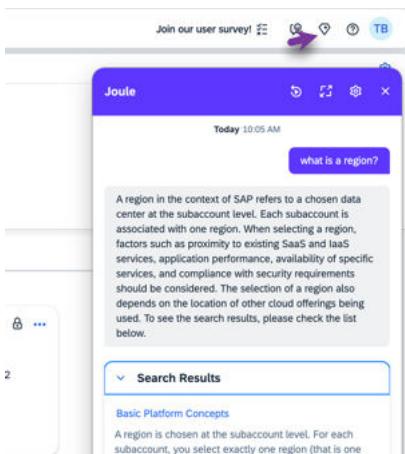
This offering is a beta feature. Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. For more information, see [Important Disclaimers and Legal Information](#).

Take your SAP BTP cockpit administration questions to Joule. Just type a question in the Joule chat using natural language and let Joule scan the product documentation on the SAP Help Portal for answers.

Joule summarizes the relevant SAP Help Portal information. This summary is then displayed along with the top help results. If you select a link result, you're taken directly to the topic on the SAP Help Portal.

#### *ⓘ* Note

Joule is an addition to in-app help content, guided tours, embedded learning content, and customer-specific enablement content in SAP Companion.



## Related Information

[Features of Joule](#)

[Supported Administrative Tasks \(Beta\) \[page 2319\]](#)

### 6.1.2.4 Troubleshooting Joule in SAP BTP cockpit

When you work with Joule, you could have questions about the things you see in the chat conversation. A list of common questions and answers are provided here.

## Subaccount Capabilities

### I don't see my subaccount in the subaccount list

- **Cause 1:** In some cases, Joule is limited to displaying the first 15 subaccounts, which are listed alphabetically.
- **Solution 1:** Type the name of your subaccount directly in the chat window.
- **Cause 2:** Joule can only list subaccounts that are successfully created in SAP BTP cockpit.
- **Solution 2:** Check your SAP BTP cockpit global account and make sure that the subaccount was created successfully. If the create action failed, you'll see a red tag on the subaccount that says *Creation Failed*.

### 6.1.3 Account Administration Using the SAP BTP Command Line Interface (btp CLI)

Use the SAP BTP command line interface (btp CLI) for all account administration tasks, such as creating or updating subaccounts, authorization management, and working with service brokers and platforms. It is an

alternative to the SAP BTP cockpit for users who like to work in a terminal or want to automate operations using scripts.

The btp CLI offers convenient features for interactive use as well as for scripting: For the interactive user, there's login via SSO, command autocompletion, as well as several interactive commands with prompts. But all commands can also be run with parameters, and the command output can be set to json format, so the btp CLI is perfect for scripting.

If you want to use the btp CLI for scripting, we recommend to use the `--format json` option or to persistently set the output format to json with `btp set config --format json`. See [Change the Output Format to JSON \[page 2350\]](#) and [Change Configuration Settings \[page 2349\]](#).

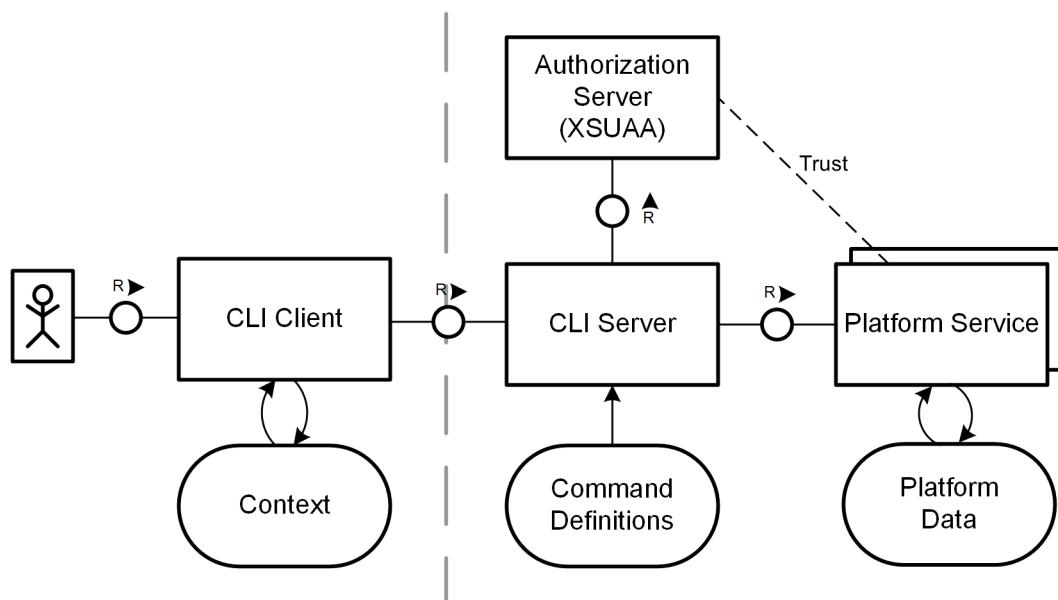
## Documentation

In this documentation, you learn how to download, install, update, and use the btp CLI. Additionally, see the following:

- [btp CLI Command Reference](#): The command reference contains descriptions of all commands with their parameters, as well as examples, tips, and links to further conceptual documentation.
- [Get Started with the SAP BTP Command Line Interface \(btp CLI\)](#): This tutorial walks you through downloading and installing the btp CLI client, understanding the syntax, and running your first commands.
- [Setting Up a Global Account via the Command Line \[page 162\]](#): This chapter lists the commands of the btp CLI and the cf CLI that you need to create a subaccount and enable Cloud Foundry.
- [What's New for SAP BTP Command Line Interface](#): Release notes for the btp CLI.

When working with the btp CLI in your terminal, you can access the help by executing `btp help` or by appending `--help` to a command.

## How the btp CLI Works



You download the btp CLI client to your local desktop and access it through the shell of your operating system. The client then accesses all required platform services through its backend, the CLI server, where the command definitions are stored. The CLI server delegates authentication and authorization to the authorization server, and forwards trust to the platform services, which then take care of authorization at the execution of each command.

## Related Information

[btp CLI Command Reference](#)

### 6.1.3.1 Download and Start Using the btp CLI Client

To use the SAP BTP command line interface (btp CLI), you need to download the client first.

## Context

Each client version is supported for at least one year and most of the updates to the btp CLI don't require a new client installation, but are made available through the btp CLI server so that you can use them in your installed version of the client. Regularly updating the client, however, will ensure that you don't miss out on any new features or security updates.

Check out the [What's New for SAP Business Technology Platform](#) page on SAP Help Portal (filtered for [SAP BTP Command Line Interface](#)) to stay informed about features, client releases, as well about client deprecations and discontinuations.

The client is available for 64-bit versions of the following operating systems:

- Microsoft Windows (amd64)
- Apple macOS (amd64 and arm64)
- Linux (amd64 and arm64)

There are various ways of downloading the client for each operating system, as explained below.

## Related Information

[How to Work with the btp CLI \[page 2333\]](#)

[Command Syntax of the btp CLI \[page 2329\]](#)

[Set a Target for Subsequent Commands with btp target \[page 2346\]](#)

[btp CLI Command Reference](#)

## 6.1.3.1.1 Microsoft Windows (amd64)

### Procedure

1. You can get the btp CLI client by running the installation script or manually downloading it.
  - Option 1: **Installation Script (Windows PowerShell)**  
When you run the installation script in Windows PowerShell, you will be guided through the installation process:

```
Invoke-RestMethod 'https://cli.btp.cloud.sap/btpcli-install.ps1' | Invoke-Expression
```

To uninstall the btp CLI:

```
Invoke-RestMethod 'https://cli.btp.cloud.sap/btpcli-uninstall.ps1' | Invoke-Expression
```
  - Option 2: **Manually install the latest Microsoft Windows (amd64) version**
    1. Download the latest [Microsoft Windows client](#) from [SAP Development Tools](#). If you prefer using the command line, use the curl command, which downloads and saves the latest client version to your computer and accepts the SAP Developer Agreement (EULA).

```
curl -LJO https://tools.hana.ondemand.com/additional/btp-cli-windows-amd64-latest.tar.gz --cookie "eula_3_2_agreed=tools.hana.ondemand.com/developer-license-3_2.txt"
```
    2. Extract the client from tar.gz archive.  
Use PowerShell or an external program, such as WinRAR, to extract the tar.gz file. Tip: Once you've unpacked the executable file, enter cmd or powershell in the address bar of the folder. This opens the command prompt or PowerShell in this folder. Or proceed with the next step in order to enable calling btp from any location on your system.
    3. Copy the client executable from the unpacked folder to a directory of your choice. We recommend the following, because it ensures that you can call btp system-wide, i.e., that it is in your PATH:  
`C:/Users/<your-user>`  
In the Windows search, enter [Environment Variable](#) and open the [System Properties](#). On the [Advanced](#) tab, open [Environment Variables](#). Under [User Variables](#), open the [Path](#) entry and add the file location of the btp.exe (`c:/Users/<your-user>`).
    4. **Optional:** If you have a proxy server configured in your environment, you need to specify its address and port as environment variable HTTPS\_PROXY or https\_proxy to access SAP BTP. For example, `HTTPS_PROXY=https://my-https-proxy:1234`.  
The specified proxy server is then used for HTTPS requests, unless overridden by the NO\_PROXY or no\_proxy environment variables, which define a comma-separated list of hosts to be excluded from proxying.
2. Run btp in your terminal. Note that you need read and write permissions in the target folder to run this executable.
3. Log into your global account with `btp login` or `btp login --sso`. The login flow is interactive and will prompt for all required information. Note that you need to confirm the btp CLI server (`https://cli.btp.cloud.sap/` and, if you have more than one, select your global account. For details, see [Log in](#).

- Once you're logged in, familiarize yourself with the btp CLI, for example with [How to Work with the btp CLI](#), [Command Syntax of the btp CLI](#), or simply by trying out a few commands, such as the following:

```
btp list accounts/subaccount  
btp list security/user  
btp get security/user "name@example.com"  
btp get accounts/global-account"
```

→ Tip

Use the command completion feature in the btp CLI to save keystrokes when entering command actions, group-object combinations, and their parameters in the command line. For more information, see [Enable Command Autocompletion \[page 2344\]](#).

- If you're going to work in a subaccount of this global account, consider setting the target to this subaccount using `btp target` and select the subaccount. See [Set a Target for Subsequent Commands with btp target](#).
- To view the current context, target, and version, use `btp`.

### 6.1.3.1.2 Apple macOS (amd64 and arm64)

#### Procedure

- You can get the btp CLI client by running the installation script or manually downloading it.

- Option 1: **Installation Script**

When you run the installation script in your Apple macOS terminal, you will be guided through the installation process:

```
curl https://cli.btp.cloud.sap/btpcli-install.sh | zsh
```

To uninstall the btp CLI:

```
curl https://cli.btp.cloud.sap/btpcli-uninstall.sh | zsh
```

- Option 2: **Manually install the latest Apple macOS (amd64 and arm64) version**

- Download the latest Apple macOS client (`amd64` or `arm64`) from [SAP Development Tools](#). If you prefer using the command line, use the curl command, which downloads and saves the latest client version to your computer and accepts the SAP Developer Agreement (EULA).

For Apple macOS amd64:

```
curl -LJO https://tools.hana.ondemand.com/additional/btp-cli-darwin-amd64-latest.tar.gz --cookie "eula_3_2_agreed=tools.hana.ondemand.com/developer-license-3_2.txt"
```

For Apple macOS arm64:

```
curl -LJO https://tools.hana.ondemand.com/additional/btp-cli-darwin-arm64-latest.tar.gz --cookie "eula_3_2_agreed=tools.hana.ondemand.com/developer-license-3_2.txt"
```

2. Extract the client from tar.gz archive.  
Open the tar.gz file by double-clicking it.
3. Copy the client executable from the unpacked folder to a directory of your choice. We recommend the following, because it ensures that you can call `btp` system-wide, i.e., that it is in your PATH:  
`/usr/local/bin`
4. **Optional:** If you have a proxy server configured in your environment, you need to specify its address and port as environment variable `HTTPS_PROXY` or `https_proxy` to access SAP BTP. For example, `HTTPS_PROXY=https://my-https-proxy:1234`.  
The specified proxy server is then used for HTTPS requests, unless overridden by the `NO_PROXY` or `no_proxy` environment variables, which define a comma-separated list of hosts to be excluded from proxying.

2. Run `btp` in your terminal. Note that you need read and write permissions in the target folder to run this executable.
3. Log into your global account with `btp login` or `btp login --sso`. The login flow is interactive and will prompt for all required information. Note that you need to confirm the `btp` CLI server (`https://cli.btp.cloud.sap/`) and, if you have more than one, select your global account. For details, see [Log in](#).
4. Once you're logged in, familiarize yourself with the `btp` CLI, for example with [How to Work with the `btp` CLI](#), [Command Syntax of the `btp` CLI](#), or simply by trying out a few commands, such as the following:

```
btp list accounts/subaccount
```

```
btp list security/user
```

```
btp get security/user "name@example.com"
```

```
btp get accounts/global-account"
```

#### → Tip

Use the command autocompletion feature in the `btp` CLI to save keystrokes when entering command actions, group-object combinations, and their parameters in the command line. For more information, see [Enable Command Autocompletion \[page 2344\]](#).

5. If you're going to work in a subaccount of this global account, consider setting the target to this subaccount using `btp target` and select the subaccount. See [Set a Target for Subsequent Commands with `btp target`](#).
6. To view the current context, target, and version, use `btp`.

### 6.1.3.1.3 Linux (amd64 and arm64)

#### Procedure

1. You can get the `btp` CLI client by running the installation script or manually downloading it.

- Option 1: **Installation Script**

When you run the installation script in your Linux terminal, you will be guided through the installation process:

```
curl https://cli.btp.cloud.sap/btpcli-install.sh | bash
```

To uninstall the btp CLI:

```
curl https://cli.btp.cloud.sap/btpcli-uninstall.sh | bash
```

- Option 2: **Manually install the latest Linux (amd64 and arm64) version**

1. Download the latest Linux client ([amd64](#) or [arm64](#)) from [SAP Development Tools](#). If you prefer using the command line, use the curl command, which downloads and saves the latest client version to your computer and accepts the SAP Developer Agreement (EULA).

For Linux amd64:

```
curl -LJO https://tools.hana.ondemand.com/additional/btp-cli-linux-amd64-latest.tar.gz --cookie "eula_3_2_agreed=tools.hana.ondemand.com/developer-license-3_2.txt"
```

For Linux arm64:

```
curl -LJO https://tools.hana.ondemand.com/additional/btp-cli-linux-arm64-latest.tar.gz --cookie "eula_3_2_agreed=tools.hana.ondemand.com/developer-license-3_2.txt"
```

2. Extract the client from tar.gz archive.

Use the terminal to open the tar.gz file with `tar -xzf btp-cli-linus-amd64-latest.tar.gz`

3. Copy the client executable from the unpacked folder to a directory of your choice. We recommend the following, because it ensures that you can call `btp` system-wide, i.e., that it is in your PATH:  
`/usr/local/bin`

4. **Optional:** If you have a proxy server configured in your environment, you need to specify its address and port as environment variable `HTTPS_PROXY` or `https_proxy` to access SAP BTP. For example, `HTTPS_PROXY=https://my-https-proxy:1234`.  
The specified proxy server is then used for HTTPS requests, unless overridden by the `NO_PROXY` or `no_proxy`, environment variables, which define a comma-separated list of hosts to be excluded from proxying.

2. Run `btp` in your terminal. Note that you need read and write permissions in the target folder to run this executable.
3. Log into your global account with `btp login` or `btp login --sso`. The login flow is interactive and will prompt for all required information. Note that you need to confirm the btp CLI server (`https://cli.btp.cloud.sap/`) and, if you have more than one, select your global account. For details, see [Log in](#).
4. Once you're logged in, familiarize yourself with the btp CLI, for example with [How to Work with the btp CLI](#), [Command Syntax of the btp CLI](#), or simply by trying out a few commands, such as the following:

```
btp list accounts/subaccount
```

```
btp list security/user
```

```
btp get security/user "name@example.com"
```

```
btp get accounts/global-account"
```

→ Tip

Use the command completion feature in the btp CLI to save keystrokes when entering command actions, group-object combinations, and their parameters in the command line. For more information, see [Enable Command Autocompletion \[page 2344\]](#).

5. If you're going to work in a subaccount of this global account, consider setting the target to this subaccount using `btp target` and select the subaccount. See [Set a Target for Subsequent Commands with btp target](#).
6. To view the current context, target, and version, use `btp`.

### 6.1.3.1.4 Get Updates

Updating the btp CLI client is essentially replacing the old executable file with a newer version.

#### Context

Each btp CLI client version is supported for at least a year after its release, but we recommend to regularly update the client. This way, you won't miss any new features or security updates. The [What's New for SAP Business Technology Platform](#) page on SAP Help Portal announces new features and informs you if you need a client update to use them. If your client version is deprecated, an update hint is displayed in the client as well.

To find out the version of the CLI client you are using, run `btp --info` or simply `btp`.

#### Procedure

1. Get the latest version either by re-running the installation script or downloading the client manually.
2. Extract the client file (for example: `btp.exe`) and replace the old file with this new one.
3. Work with the btp CLI as usual and enjoy the new features.

#### Related Information

[Log in \[page 2337\]](#)

[Commands in the btp CLI \[page 2354\]](#)

## 6.1.3.2 Command Syntax of the btp CLI

Each command consists of the base call `btp` followed by a verb (the action), a combination of group and object, and parameters.

The `btp` CLI uses the following syntax:

```
btp [OPTIONS] ACTION GROUP/OBJECT [PARAMS]
```

The commands are ordered in groups and you need to specify the object on which you want to carry out an action by the group/object combination. Words in caps are placeholders, and brackets [ ] denote optionality. Optionally, you can pass options with a command call. Here's is one example with the verbose option and no parameters before we outline the entire syntax:

### ↔ Sample Code

```
btp --verbose list accounts/subaccount
```

- `btp` is the base call to start each command
  - `OPTIONS` can be added to each command, for example `--format json` to change the output format to `json`, or `--verbose` to execute a command in verbose mode. For details, see [How to Work with the btp CLI \[page 2333\]](#).
  - `ACTION` is the verb. Depending on the `GROUP/OBJECT` combination, different verbs are available, such as `get`, `list`, `create`, `delete`, `assign`, `unassign`, `add`, `remove`. For a complete list, use `btp help`. To find out which commands are available for a specific action, use `btp help ACTION`, for example, `btp help list`.
  - Special `ACTION`: the `help ACTION`. You can always place `help` at the beginning of a command and still add further parts of the command, such as the `ACTION` or `GROUP` or `GROUP/OBJECT` combination for which you want to call help. See [Get Help \[page 2336\]](#).
  - The `GROUP/OBJECT` combination specifies the entity that the action is carried out on. For example, all commands related to users and their authorizations belong to the **security** group, in which **objects** such as `role`, `role-collection`, and `user` are available. There are currently three groups:
    - `accounts`: Objects related to the account model, subscriptions, and environments
    - `security`: Authorization objects and users
    - `services`: Objects related to SAP Service Manager
- To get help on a particular group, use `btp help GROUP`, for example `btp hep accounts`. This will display all objects and related actions available in that group.
- `PARAMETERS` are passed with most commands. With `btp login`, for example, you don't have to pass parameters up front, but you'll be prompted to enter them. The same applies to the `btp target` command. And `btp logout` does not need parameters as it will log out the current user from the global account. Some commands have one positional parameter, which is entered directly after the command. All further parameters have a key and can be optional. The command `help` specifies the optional parameters as such. For example, in `btp assign security/role-collection "Global Account Administrator" --to-user example@mail.com --of-idp my-idp`, "Global Account Administrator" is the positional parameter, and the other two parameters have keys.

### Note

The commands that you type into the command line are interpreted and executed by the shell. Make sure you're familiar with your shell to avoid unexpected interferences. For examples of correct escaping, see [Passing JSON Parameters on the Command Line \[page 2331\]](#).

### Tip

You can use the command completion feature in the btp CLI to save keystrokes when entering commands actions, group-object combinations, and their parameters in the command line. For more information, see [Enable Command Autocompletion \[page 2344\]](#).

## A Few Commands to Get Started

Here are a few commands for you to try out once you're logged in ([Log in \[page 2337\]](#)):

```
btp list accounts/subaccount  
btp list security/user  
btp get security/user "name@example.com"  
btp list accounts/subscription
```

## Example

In this example, we assign the **Global Account Administrator** role collection to user `name@example.com` and try out some options.

Use `help` or `--help` to display command-specific help to learn how to use the command:

```
btp help assign security/role-collection  
btp assign security/role-collection --help  
btp --help assign security/role-collection
```

Command with positional parameter and one mandatory parameter:

```
btp assign security/role-collection "Global Account Administrator" --to-user  
"name@example.com"
```

Command with positional parameter, mandatory parameter, and optional parameter:

```
btp assign security/role-collection "Global Account Administrator" --to-user  
"name@example.com" --of-idp "my-idp"
```

The same command in verbose mode:

```
btp --verbose assign security/role-collection "Global Account Administrator" --  
to-user "name@example.com" --of-idp "my-idp"
```

## Related Information

[btp CLI Command Reference](#)

### 6.1.3.2.1 Passing JSON Parameters on the Command Line

Depending on the shell you use, the escaping rules are different. Here you'll find examples of correct escaping and quotes when passing JSON objects on the command line using different shells.

The table below gives an overview of the escaping rules in these commonly used shells. VALUE refers to one single line in json.

Shell	Correct Escaping and Quotes
Bash	Use <code>--parameter "VALUE"</code> and escape quotes with \ " within VALUE or Use <code>--parameter 'VALUE'</code> and do not escape quotes within VALUE
Windows Command Prompt	Use <code>--param "VALUE"</code> and escape quotes with \ " within VALUE
Windows PowerShell	Use <code>--param "VALUE"</code> and escape quotes with \ " within VALUE

#### → Tip

The CLI client provides examples in the command help for all commands. The examples that include JSON parameters use formatting that is compliant with Bash (Unix/Linux operating system, macOS) and Windows Command Prompt.

The command-line examples in this chapter are based on the following JSON:

```
{  
    "Key1": [ "Value1" ],  
    "Key2": [ "Value1", "Value2" ]  
}
```

## Bash (Linux, macOS)

Option 1

```
--parameter "{\"Key1\":[\"Value1\"],\"Key2\":[\"Value1\", \"Value2\"]}"
```

For example, when creating a subaccount (`btp create accounts/subaccount`) with labels `Department = Sales` and `Contacts = name1@example.com` and `name2@example.com`, use the following syntax:

↳ Sample Code

```
btp create account/subaccount --display-name "my-subaccount" --region us10  
--subdomain "my-subdomain" --labels "{\"Department\":[\"Sales\"],\"Contacts\":[\"name1@example.com\", \"name2@example.com\"]}"
```

Option 2

```
--parameter '{"Key1": ["Value1"], "Key2": ["Value1", "Value2"]}'
```

For example, when creating a subaccount (`btp create accounts/subaccount`) with labels `Department = Sales` and `Contacts = name1@example.com` and `name2@example.com`, use the following syntax:

↳ Sample Code

```
btp create account/subaccount --display-name "my-subaccount" --region us10  
--subdomain my-subdomain --labels {"Department": ["Sales"], "Contacts":  
["name1@example.com", "name2@example.com"]}
```

## Windows Command Prompt

```
--parameter "{\"Key1\":[\"Value1\"],\"Key2\":[\"Value1\", \"Value2\"]}"
```

For example, when creating a subaccount (`btp create accounts/subaccount`) with labels `Department = Sales` and `Contacts = name1@example.com` and `name2@example.com`, use the following syntax:

↳ Sample Code

```
btp create account/subaccount --display-name "my-subaccount" --region us10  
--subdomain "my-subdomain" --labels "{\"Department\":[\"Sales\"],\"Contacts\":[\"name1@example.com\", \"name2@example.com\"]}"
```

## Windows PowerShell

```
--parameter '{"Key1\":[\"Value1\"],[\"Key2\":[\"Value1\", \"Value2\"]]}'
```

For example, when creating a subaccount (`btp create accounts/subaccount`) with labels `Department = Sales` and `Contacts = name1@example.com` and `name2@example.com`, use the following syntax:

#### ↳ Sample Code

```
./btp create account/subaccount --display-name "my-subaccount" --region us10  
--subdomain my-subdomain --labels '{\"Department\": [\"Sales\"], \"Contacts\":  
[\"name1@example.com\", \"name2@example.com\"]}'
```

## Passing JSON Parameters as Files

You can also transfer JSON parameters by providing the full path to a JSON file. For example:

```
btp create services/binding --name my-binding-name --instance-name my-service-  
instance-name --parameters "<your-user-directory>/Documents/parameters.json"
```

## Related Information

[Command Syntax of the btp CLI \[page 2329\]](#)

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)

[Working with Environments Using the btp CLI \[page 2360\]](#)

### 6.1.3.3 How to Work with the btp CLI

Learn how to work with the SAP BTP command line interface (btp CLI). For example, how to log in, get help, and set a default context for commands.

## General Commands

```
btp help
```

```
btp feedback
```

```
btp login
```

```
btp logout
```

```
btp target
```

```
btp enable autocomplete
```

```
btp disable autocomplete
```

```
btp list config
```

```
btp set config
```

```
btp reset config
```

## Options for Each Command

The following options are available for each command. They need to be typed right after the base call `btp`, and they can be combined (for example, `btp --verbose --help list accounts/subaccount`). The `--help` option also works at the end of a command call.

```
btp [OPTIONS] ACTION GROUP/OBJECT [PARAMS]
```

### Options

--config	Specifies the location of the configuration file. See <a href="#">Specify the Location of the Configuration File [page 2352]</a> .
--info	Displays client and server versions, target, and context. Note that this option only works on its own ( <code>btp --info</code> ) and cannot be added to other command calls. You can also just use <code>btp</code> to display this info. See <a href="#">View Version and Current Context [page 2337]</a> .
--help	Displays help. See <a href="#">Get Help [page 2336]</a> .

## Options

---

--verbose	<p>Prints tracing information for support or for your own troubleshooting. See <a href="#">Troubleshooting and Support [page 2375]</a>.</p> <p>To set the command output to verbose persistently, you can change the configuration settings with <code>btp set config --verbose true</code>. See <a href="#">Change Configuration Settings [page 2349]</a>.</p>
	<p><b>⚠ Caution</b></p> <p>The <code>--verbose</code> option prints input data that you pass as parameters through files. For example, if you pass a <code>.json</code> file as a parameter without the <code>--verbose</code> option, its content is sent directly to the backend service to be processed. If you use the <code>--verbose</code> option, the <code>btp</code> CLI prints the content of the file to the terminal. If there is sensitive information in such a file, the <code>btp</code> CLI cannot filter it out, as it doesn't understand its semantics.</p> <p>Use the verbose output only if you can control the output of your terminal, i.e. only locally. It should not be used in production or shared environments, where logs may be persisted or even transported out of the system.</p>
--format	<p>Changes the output format of a command to JSON. See <a href="#">Change the Output Format to JSON [page 2350]</a>.</p> <p>To set the command output to json persistently, you can change the configuration settings with <code>btp set config --format json</code>. See <a href="#">Change Configuration Settings [page 2349]</a>.</p>
--version	Prints the version of the client.

## Related Information

[btp CLI Command Reference](#)

## 6.1.3.3.1 Get Help

There is extensive help in the btp CLI about every command. You can get help with the `help` action or the `--help` option.

There are several ways to call help in the btp CLI, and there is help on all levels, from a basic entry page that explains the syntax to very detailed command-specific help that explains the usage of a command and its parameters in detail.

The easiest way to get help is probably the `help` action (for basic help, use `btp help`, as shown in the examples below. But you can also place `--help` at the end or beginning of a command call, for example, `btp list help`, `btp --help list`, and `btp list --help` are interchangeable).

Here's the help syntax:

- `btp help`
- `btp help <ACTION>`
- `btp help <GROUP>`
- `btp help <GROUP>/<OBJECT>`

Example Help Calls (progressing in level of detail)	What to Expect
<code>btp help</code>	Explains the syntax and shows plenty of examples.
<code>btp help all</code>	Displays an overview of all available commands, ordered according to their group/object combinations.
<code>btp help list</code>	Displays <code>list</code> commands with short summaries.
<code>btp help accounts</code>	Displays all objects in <code>accounts</code> group, and all available actions per group/object combination.
<code>btp help accounts/subaccount</code>	Displays all available commands for this group/object combination.
<code>btp help list accounts/subaccount</code>	Displays command-specific help, such as usage, a list of all parameters with descriptions, helpful tips, examples, and links to further documentation.

## Related Information

[Command Syntax of the btp CLI \[page 2329\]](#)

## 6.1.3.3.2 View Version and Current Context

To find out the current context you're working in, run the command `btp --info` or simply `btp`.

### Procedure

Use `btp --info` or just `btp`.

The client displays its own version, usage information, the CLI server URL and version, the current user, the location of the configuration file, as well as the global account, directory, or subaccount you're targeting in their account hierarchy.

### Related Information

[Set a Target for Subsequent Commands with `btp target` \[page 2346\]](#)

## 6.1.3.3 Log in

Log in with the `btp` CLI is on global account level.

### Prerequisites

- You need to pass the CLI server URL. Usually, it is proposed during login and you can confirm with `ENTER`:  
`https://cli.btp.cloud.sap/`.  
Note that in clients up until version 2.49.0, the old server URL `https://cpcli.cf.eu10.hana.ondemand.com` is proposed. For the time being, you can use either one, but we recommend to switch to `https://cli.btp.cloud.sap/`. If you are in a private cloud and your operator has provided you with a different server URL, you'll have to enter that one.
- Your user is assigned to the Global Account Viewer or the Global Account Administrator role collection. See [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).
- To log on with SAP Universal ID, you need to log on with the single sign-on parameter (`btp login --sso`). If you don't want to use single sign-on, log on with the password associated with your account (S-user or P-user) in the default identity provider, SAP ID service. If you've forgotten this password and this user is associated with your SAP Universal ID user, reset your password.  
For more information, see SAP Note [3085908](#) and [Log in with Single Sign-On \[page 2341\]](#).

## Context

When you log in with the btp CLI, a session token is saved in your local configuration. This allows you to keep your authentication over multiple command calls or after restarting your computer. The maximum session length is 12 hours. After that, you'll have to log in again. If you want to end your login earlier, use `btp logout`.

### Login with single sign-on

We recommend using single sign-on and authenticating directly at your identity provider through a web browser:

```
btp login --sso
```

See [Log in with Single Sign-On \[page 2341\]](#).

### Login with a custom identity provider

To login with a custom identity provider, use:

```
btp login --sso --idp <TENANT>
```

See [Log in with a Custom Identity Provider \[page 2342\]](#).

## Procedure

To log in manually, use `btp login`. The btp CLI prompts for all login information, but optionally, you can provide the required information as parameters.

Usage: `btp [OPTIONS] login [PARAMS]`

Parameters

<code>--sso</code>	Opens a browser for single sign-on at the identity provider. The btp CLI doesn't prompt for this parameter. If you use SAP Universal ID, you need to use this parameter.
	To suppress automatic browser opening, use <code>--sso manual</code> .

--idp <TENANT>

This parameter is only needed if you want to log in through a custom identity provider. The CLI doesn't prompt for it.

If trust is configured between your global account and a custom identity provider, use this parameter to log in through this identity provider by providing its tenant ID. To retrieve the required value, you can run `btp list security/trust` and use the value from the *Tenant* column. Or you can use the Global Account view in the cockpit, under *Security → Trust Configuration → Custom Platform Identity Providers*. The value is displayed in the *BTP CLI* column.

#### ⓘ Note

To work with users from a custom identity provider, you need to specify the `--of-idp` parameter by providing the origin key of the custom identity provider. This is applicable to the following commands: `btp list security/user`, `btp get security/user`, `btp delete security/user`, `btp assign security/role-collection`, `btp unassign security/role-collection`, and you find this origin key in the cockpit under *Security*.

For more information about using a custom identity provider, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).

---

--url <URL>

The client proposes this CLI server URL: `https://cli.btp.cloud.sap/`, which you can confirm by pressing **[ENTER]**. If your operator has provided you with a different server URL, you can specify it here. Note that when you enter a new server URL for the first time, you're asked to confirm that you trust it.

---

--user <USER>

Your user name, usually an email address.

---

--password <PASSWORD>

Your password. If you have enabled Two-Factor-Authentication, append your token to your password.

#### → Tip

We don't recommend to provide the password with this parameter, as it appears in plain text and may be recorded in your shell history. Rather, enter it when you're prompted.

```
--subdomain <GLOBALACCOUNT>
```

In the interactive login, after successful authentication, the btp CLI will offer all of your global accounts so you can select the one to log in to.

You can also provide the global account as a parameter by specifying its subdomain. You should have obtained the subdomain from your operator; but you can also find it in the cockpit in the global account view.

 Trial Home /  1234567xtrial

#### Global Account: 1234567xtrial - Account Explorer

Filtered: 0 of 2 directories, 2 of 2 subaccounts | Subdomain: 1234567xtrial-ga

```
btp login
```

If you've logged in before, the server URL, the subdomain, and the user from the last login are suggested. You can then press **Enter** to confirm, or type in different values.

```
CLI server URL [https://cli.btp.cloud.sap/]>
Subdomain [my-global-account]>
User [name@example.com]>
```

## Results

Upon successful login, the btp CLI creates a folder (`btp`) and a configuration file (`config.json`) in the default location of your user data directory:

- Microsoft Windows: `C:\Users\<username>\AppData\Roaming\SAP\btp\config.json`
- Apple macOS: `~/Library/Application Support/.btp/config.json`
- Linux: `~/.config/.btp/config.json`

To change this location, use the `--config` option or the environment variable. See [Specify the Location of the Configuration File \[page 2352\]](#).

### → Tip

You've logged in to the global account and all commands are executed in this global account, unless you provide a subaccount or directory ID with the command. To change this default context for subsequent commands, you can target a subaccount or directory, or even a different global account by using `btp target`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

## Related Information

[btp CLI Command Reference](#)

### 6.1.3.3.1 Log in with Single Sign-On

Use the single sign-on parameter (`btp login --sso`) to log in to your identity provider through a browser instead of passing username and password on the command line.

#### Prerequisites

Login through a browser is only available if you use the default server URL that is proposed by the CLI during `login (https://cli.btp.cloud.sap/ )`. This is the case unless your operator has provided you specifically with a different server URL.

#### Context

If you have a custom identity provider configured in your global account, use `btp login --sso --idp <TENANT>`. See [Log in with a Custom Identity Provider \[page 2342\]](#).

##### → Tip

If you like logging in through single sign-on in a browser, you can configure this as the default behavior of `btp login` to save a few keystrokes at login. Use `btp set config --login.sso browser`. See [Change Configuration Settings \[page 2349\]](#).

#### Procedure

1. Enter the following and press `ENTER` to be prompted for the server URL and the subdomain of your global account:

```
btp login --sso
```

Alternatively, you can pass the server URL and the subdomain of your global account as parameters:

```
btp login --sso --url https://cli.btp.cloud.sap/ --subdomain <my-subdomain>
```

To suppress the automatic browser-opening, you can run the command as follows:

```
btp login --sso manual
```

2. Authenticate directly at your identity provider through your browser.

If you already have an active session, the identity provider immediately transfers a token to the client, which completes your login. Otherwise, you need to enter your credentials.

## Results

You've successfully logged in. You can return to the command line to work in your global account or - at first login - to select the global account for login.

## Related Information

[Log in \[page 2337\]](#)

[Log in with a Custom Identity Provider \[page 2342\]](#)

### 6.1.3.3.3.2 Log in with a Custom Identity Provider

## Context

If trust is configured between your global account and a custom identity provider, you need to use the `--idp` parameter to log in through this identity provider. As value, you provide its tenant ID. To retrieve the required value, you can use `btp list security/trust` and check the *Tenant* column. Or you can use the Global Account view of the cockpit under [Security → Trust Configuration → Custom Platform Identity Providers](#) and use the value from the *BTP CLI* column.

#### ⓘ Note

To work with users from a custom identity provider, you need to specify the `--of-idp` parameter by providing the origin key of the custom identity provider. This is applicable to the following commands: `btp list security/user`, `btp get security/user`, `btp delete security/user`, `btp assign security/role-collection`, `btp unassign security/role-collection`, and you find this origin key in the cockpit under [Security](#).

To learn how to configure trust to a custom identity provider, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#) or the command help of `btp create security/trust`.

#### ⚠ Restriction

Keep in mind that each user is allowed a maximum of 10 parallel sessions per identity provider. This number takes into account all tools, including the cockpit and CLIs.

For more information, see [Restrictions When Using Custom Identity Providers for Platform Users \[page 2243\]](#).

## Procedure

1. To make use of single sign-on, log in with:

```
btp login --sso --idp <TENANT>
```

2. To provide user and password on the command line, log in with:

```
btp login --idp <TENANT>
```

You will be prompted for all required login information.

## Related Information

[Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#)

[Trust and Federation with Identity Providers \[page 2204\]](#)

[Log in \[page 2337\]](#)

### 6.1.3.3.4 Log out

Logging out of the configured server removes all user-specific data from the configuration file.

## Context

Once you're finished using the btp CLI and you want to ensure that your locally stored credentials are immediately deleted, you can run the logout command. If you choose not to log out, your credentials will expire 24 hours after you last command execution, but the next time you log in, the btp CLI will propose your current subdomain and user so you won't have to type it in again.

## Procedure

To log out, use `btp logout`.

This terminates your active logout session and ensures that all user-specific data is removed. The next time you log in, you will have to type in the subdomain of the global account and your user.

## Related Information

[Log in \[page 2337\]](#)

## 6.1.3.3.5 Enable Command Autocompletion

Use command autocompletion to save keystrokes when entering command actions, group-object combinations, and their parameters in the SAP BTP command line interface (btp CLI).

### Context

Autocompletion in btp CLI currently supports the following shells:

- Bash
- PowerShell
- Zsh

#### ⓘ Note

The respective shell must be installed on your operating system before enabling autocomplete.

Once autocomplete is enabled (it's disabled by default), you use the autocomplete feature as follows in the command line:

- Enter a partial command action, group-object combination, or parameter, and then press the `Tab` key. The command line either automatically completes your command or, when there's more than one option available, it displays a list of suggested command actions/options/parameters.
- When a suggestion list is displayed, depending on your shell, use the `Tab` or arrow keys to move through the list and press `Enter` to make a selection.

### Examples

The following examples show various ways that you can use autocompletion:

- Enter `btp` and press `Tab` to display all available actions:

```
./btp TAB
add      create      enable      list      logout      register
subscribe  unassign   unsubscribe
assign     delete      get        login      move       remove
target     unregister update
```

- Partially enter `btp cre` and press `Tab` to autocomplete the command to `btp create`. Then, press `Tab` again to display a suggested list of group-object combinations:

```
./btp create TAB
accounts/directory           accounts/resource-provider    security/
role                      services/binding
accounts/environment-instance accounts/subaccount          security/role-
collection      services/instance
```

- Partially enter a group and press `Tab` to display a suggested list of objects:

```
./btp create accountsTAB
accounts/directory           accounts/environment-instance accounts/
resource-provider      accounts/subaccount
```

- Partially enter a parameter and press **Tab** to display a suggested list of parameters:

```
./btp create accounts/subaccount -TAB
--beta-enabled
--custom-properties
--description
--directory
--display-name
--global-account
--region
--subaccount-admins
--subdomain
--used-for-production
```

## Procedure

1. Use `btp enable autocomplete <SHELL>` to enable command completion for a specified shell.

The valid values for the supported shells are:

- bash
- powershell
- zsh

### ↔ Sample Code

```
btp enable autocomplete zsh
```

2. The client asks for confirmation to install the autocomplete plugin script at the specified location. Enter "y" or "yes" to continue.

Apart from the script being added to your file system, the RC file of your shell is modified to call this script at startup. The client looks for this RC file in your system and proposes possible files. You can choose to accept a proposal or specify a different RC file. If no RC file is found, the client prints an error message with the full path of the missing file, so you can create it and run the enable command again.

3. Enter the option of your choice.

To specify a custom path, first choose "Custom" and then enter the full path of the RC file to use.

For example:

```
./btp enable autocomplete powershell
This will install the autocomplete plugin script for powershell to
C:\AppData\ btp\autocomplete\scripts. Do you want to continue? [no]>y
Which RCfile should be used for the installation?
1: C:\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1
2: Custom
Enter option>2
Enter the full path of
your RCfile>C:\Documents\WindowsPowerShell\Microsoft.PowerShell_my-custom-
profile.ps1
```

4. Start a new terminal session to activate the installed autocomplete script.

## Results

When you enable command autocomplete, a script containing all the autocomplete commands is downloaded and installed in your file system.

The autocomplete option remains enabled in future sessions in your current client, until you disable it. To disable command autocomplete and uninstall the autocomplete script, run the following command:

```
btp disable autocomplete <SHELL>
```

You can run either `btp` or `btp --info` to see if command autocomplete is currently enabled and where the autocomplete script for your shell is located. If you don't see a line specifying the location of the autocomplete script, then it's disabled.

### → Tip

If you see a discrepancy between the version of the autocomplete script and the client, the update of the autocomplete script might have failed. In such a case, try to disable and enable the autocomplete feature again.

Whenever you start a new `btp` CLI terminal session, the installed autocomplete scripts are automatically updated to include the latest commands. If a script is updated, you're prompted to restart your terminal session to load the newest autocomplete information.

If disabling the command autocomplete fails or you have uninstalled the `btp` CLI client without disabling autocomplete, you can manually remove traces of the autocomplete installation in your shell initialization file (RC or profile file depending on your shell) by deleting the line that starts with `SAPCP_CLI_AUTOCOMPLETE`.

### 6.1.3.3.6 Set a Target for Subsequent Commands with `btp target`

Set the target for command calls to a subaccount, a directory, or a global account with the `btp target` command.

## Context

During login, the `btp` CLI asks you for the global account you want to log in to and this global account is then the target. This means that commands are executed in this global account unless you specify otherwise via a parameter. There are two typical use cases for the target command:

- If you know that you need to work in a particular subaccount or directory, you can change the target, so that you won't have to specify that subaccount's or directory's ID as a parameter with every command call.
- If you want to switch to another global account, you can do so by setting the target to this global account without logging in again.

By setting the target to a subaccount or directory, you also target its parent entities, that is, the **target hierarchy**. If no parameter is specified in a command call, the `btp` CLI will execute the command in the

innermost targeted entity where a command is applicable. If a subaccount is targeted that is inside a directory and the command is not available on subaccount level, it will be executed in the parent directory (if it's a directory command) or in the parent global account (if it's a global account command). This can be useful, for example, if you have targeted a subaccount, but want to update its parent directory. You can then execute `btp update accounts/directory` without specifying the directory ID.

To see the current target, use `btp --info` or simply `btp`.

To explicitly execute a command in a parent entity instead of in the target, you can specify this with '`-dir`' or '`-ga`' parameters without a value. The value is then taken from the target hierarchy. This can be useful, for example, if you have targeted a subaccount, but want to list the users of the parent directory. You can then execute `btp list security/user -dir`. Note, however, that this command is only available on a directory level if the directory is enabled to manage authorizations.

Usage: `btp [OPTIONS] target [PARAMS]`

## Procedure

1. Use `btp target` to set the target for subsequent commands.
  - Run the command without parameters to see the children, siblings and parent of your current target. You can then select your target or navigate up and down.
  - Enter `--hierarchy` as parameter to see all your global accounts with their entire account hierarchies. You can then select your target from this list.

### → Tip

If you like the `--hierarchy` parameter and would like to make it the default for the target command, you can define it as a configuration setting with `btp set config --target.hierarchy true`. If you then run `btp target`, the entire hierarchy of all your global accounts is shown.

2. Enter one of the following parameters:

Parameter	Description
<code>--global-account, -ga &lt;SUBDOMAIN&gt;</code>	The subdomain of the global account to be targeted.
<code>--directory, -dir&lt;ID&gt;</code>	The ID of the directory to be targeted. You can find the directory ID by using <code>btp get accounts/global-account --show-hierarchy</code> .
<code>--subaccount, -sa &lt;ID&gt;</code>	The ID of the subaccount to be targeted. You can find the subaccount ID by using <code>btp list accounts/subaccount</code> .

## Results

The CLI client displays the current target as a hierarchy. To execute a command in the targeted entity, you can omit the corresponding parameter.

## Example

A global account can group together different directories and subaccounts that the global account administrator makes available to users. CLI commands can be executed on all levels of this account hierarchy, that is in the global account, a directory, or a subaccount, usually specified by the above-mentioned parameters. The table below shows some example commands, and explains in which account entity they can be executed and how the target mechanism can be used.

Let's look at a global account that contains a directory with a targeted subaccount. With `btp`, the current target hierarchy is displayed like this:

```
Current target:  
My Global Account (global account, subdomain: cee12xx112345-ga)  
└ My Directory (directory, ID: 371eXXXX-55XX-40XX-b3XX-e9947ed9XXXX)  
  └ My Subaccount (subaccount, ID: d8aeXXXX-74XX-49XX-89XX-f058029eXXXX)
```

Now, when you run commands without specifying an account entity as parameter, they are executed in the targeted subaccount or in one of its parents.

Example Command	Command is Available for	Command is Executed in
<code>btp get accounts/available-environment</code>	<code>--subaccount</code>	The targeted subaccount
<code>btp update accounts/directory</code>	<code>[directory] ID</code>  Note that the directory ID is a positional parameter in this command. You can omit it when a subaccount inside the directory is targeted: <code>btp update accounts/directory [directory ID]</code> .	The parent directory of the targeted subaccount
<code>btp list accounts/available-region</code>	<code>--global-account</code>	The parent global account of the targeted subaccount
<code>btp list security/user</code>	<code>--global-account</code> <code>--directory</code> <code>--subaccount</code>	The targeted subaccount
		<b>→ Tip</b> To execute this command in a parent entity, use the <code>-dir</code> or <code>-ga</code> parameter without a value.

Example Command	Command is Available for	Command is Executed in
btp list security/user --subaccount "1111XXXX-22XX-33XX-44XX-5555555XXXX" --subaccount	--global-account --directory --subaccount	The specified subaccount, which overrides the target

## Related Information

[btp CLI Command Reference](#)  
[Global Accounts \[page 94\]](#)

### 6.1.3.3.7 Change Configuration Settings

Change the configuration settings to customize the behavior of the btp CLI.

#### Context

There are a few configuration settings that you can set to customize how the btp CLI works.

These settings are saved to the configuration file, which governs how the btp CLI works. You can have more than one configuration file for working in different accounts at the same time by logging in with the `--config PATH` option and then also providing it with each command call. See [Specify the Location of the Configuration File \[page 2352\]](#).

To find out where the configuration file is stored, use `btp --info`. Use the following commands to manage your configuration settings:

#### Procedure

1. To change a setting, run:

```
btp [OPTIONS] set config [--format FORMAT] [--verbose BOOL] [--target.hierarchy BOOL]
```

---

`--format FORMAT`

The output format of commands. Valid values: text (default), json

--verbose BOOL	If set to true, command output includes tracing information for support. Valid values: false (default), true
--target.hierarchy BOOL	If set to true, <code>btp target</code> displays the full hierarchy of all global accounts. Valid values: false (default), true See <a href="#">Set a Target for Subsequent Commands with <code>btp target</code> [page 2346]</a> .
--login.sso MODE	Single sign-on mode of <code>btp login</code> . Valid values: none (default), browser, manual. For example, if you want <code>btp login</code> to always open a browser for login at your identity provider, use <code>btp set config --login.sso browser</code> .

2. To list the current settings, run:

```
btp [OPTIONS] list config
```

3. To reset individual or all settings back to their default values, run:

```
btp [OPTIONS] reset config [--format] [--verbose] [--target.hierarchy] [--all]
```

## Related Information

[Specify the Location of the Configuration File \[page 2352\]](#)

### 6.1.3.3.8 Change the Output Format to JSON

Use the `--format json` option to change the output format to JSON. This is the recommended output format for automation with the `btp` CLI.

## Context

The standard output format of the `btp` CLI is plain text, formatted in a way that an interactive user of the `btp` CLI can easily read it and therefore often limited to the most relevant information. We don't recommend using this plain text output for automatic parsing and extracting information. The text output is not stable enough, as, for example, column width or the appearance of new lines may depend on the actual content of a response.

If you're using responses of the `btp` CLI for automation, such as in scripts, we recommend the JSON output format, as JSON is a data format that can be parsed by most programming languages. The JSON output is much more stable than the plain text output and it often includes more information in the JSON response than the respective text output. Because it is optimized for post-processing, it doesn't include non-essential information such as an OK at the end of a successful call.

## → Tip

To set the command output to json persistently, you can change the configuration settings with `btp set config --format json`. This makes using the `--format json` option obsolete. See [Change Configuration Settings \[page 2349\]](#).

See this [Getting BTP resource GUIDs with the btp CLI](#) blog post for some examples.

## Procedure

Start commands with `--format json`. Currently, the only valid value is `json`.

```
btp --format json list accounts/subaccount
```

## Results

### Example with text output

```
btp list accounts/subaccount
subaccounts in global account c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX...
subaccount id: display
name: subdomain:
region: beta-enabled: parent id:
state: state
message:

88XXXX80-9844-4XX7-8XXd-beXXXX42bfb2 my-
subaccount-1 my-CF-subdomain1
us10-TEST false c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX
OK Subaccount
created.

ebXXXX3e-f3a2-4XXf-a080-3eXXXX9b9ced my-
subaccount-2 my-CF-subdomain2
us10-TEST false c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX
OK Subaccount created.
```

### Example with JSON output

```
btp --format json list accounts/subaccount
{
  "value": [
    {
      "guid": "88XXXX80-9844-4XX7-8XXd-beXXXX42bfb2",
      "displayName": "my-subaccount-1",
      "globalAccountGUID": "c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX",
      "parentGUID": "c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX",
      "parentType": "ROOT",
      "region": "us10-TEST",
      "subdomain": "my-CF-subdomain1",
      "betaEnabled": false,
      "usedForProduction": "UNSET",
      "description": null,
      "expiryDate": null,
```

```

        "state": "OK",
        "stateMessage": "Subaccount created.",
        "createdDate": 1602759737909,
        "createdBy": "name@email.com",
        "modifiedDate": 1602759737909,
        "zoneId": null
    },
    {
        "guid": "ebXXXX3e-f3a2-4XXf-a080-3eXXXX9b9ced",
        "displayName": "CF TEST",
        "globalAccountGUID": "c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX",
        "parentGUID": "c8XXXX3d-6XX7-4XXb-8dXX-89XXXX372eXX",
        "parentType": "ROOT",
        "region": "us10-TEST",
        "subdomain": "my-CF-subdomain2",
        "betaEnabled": false,
        "usedForProduction": "UNSET",
        "description": null,
        "expiryDate": null,
        "state": "OK",
        "stateMessage": "Subaccount created.",
        "createdDate": 1602760244477,
        "createdBy": "name@email.com",
        "modifiedDate": 1602760244477,
        "zoneId": null
    }
]
}

```

## Related Information

[Change Configuration Settings \[page 2349\]](#)

### 6.1.3.3.9 Specify the Location of the Configuration File

You can change the location of the configuration file by using the `--config` option or the environment variable.

## Context

Upon successful login, the btp CLI creates a configuration file (`config.json`) in the default location of your user config directory:

- Microsoft Windows: `C:\Users\<username>\AppData\Roaming\SAP\btp\config.json`
- Apple macOS: `~/Library/Application Support/.btp/config.json`
- Linux: `~/.config/.btp/config.json`

This folder serves as the working directory of the btp CLI, that is, it's necessary for its proper functioning, and is used with each command execution.

If you want the configuration file to be created in a different folder, you can use the `--config` option in your login command and then specify this location in each command call with this `--config` option.

## Procedure

1. Specify the location of the configuration file with your login command:

```
btp --config "<file path>" login
```

2. Specify this location either with the `BTP_CLIENTCONFIG` environment variable, or use the `--config` option with each subsequent command call.

### ↔ Sample Code

```
btp --config "<file path>"
```

### ⓘ Note

In version 2.14, the environment variable `BTP_CLIENTCONFIG` was introduced. If you use an older client version, you need to use `SAPCP_CLIENTCONFIG`. Although the old one is kept, we recommend to switch to `BTP_CLIENTCONFIG`.

## Example

Let's assume you want to work in two subaccounts in parallel, using two terminals. For example, with the first terminal (A), you want to work in a subaccount with ID 1000, with the second terminal (B) you want to work in a subaccount with ID 2000, and you want to list the users in each subaccount.

### Terminal A

1. Log in to your global account using the default location of the configuration file:

```
btp login
```

Run all commands as usual. The btp CLI uses the default configuration file.

2. Set the target to subaccount 1000:

```
btp target --subaccount 1000
```

3. List the users of subaccount 1000:

```
btp list security/user
```

### Terminal B

1. Log in to your global account using a different location of the configuration file:

```
btp --config "C:\my-cli-folder" login
```

Use this option with each command call.

2. Set the target to subaccount 2000:

```
btp --config "C:\my-cli-folder" target --subaccount 2000
```

3. List the users of subaccount 2000:

```
btp --config "C:\my-cli-folder" list security/user
```

## 6.1.3.3.10 Provide Feedback about the btp CLI

With `btp feedback`, you can open a feedback survey in a web browser in which you can share your opinion about the btp CLI with the development team.

### Context

A short survey will open in your web browser, in which you're asked how satisfied you are with the btp CLI, and where you can share additional comments. This feedback is carefully analyzed by the development team and a valuable source of input for further improving the btp CLI.

You can also fill it out now: [Take our survey about the btp CLI](#) 

Thanks a lot for your participation. Your feedback is greatly appreciated.

### Procedure

Open the feedback survey with the following command:

```
btp feedback
```

## 6.1.3.4 Commands in the btp CLI

Find a full reference of all btp CLI commands and their parameters here: [btp CLI Command Reference](#).

### [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

Use the SAP BTP command line interface (btp CLI) to manage operations with global accounts, directories, and subaccounts.

### [Setting Entitlements Using the btp CLI \[page 2359\]](#)

Use the SAP BTP command line interface (btp CLI) to set entitlements to define the functionality or permissions available for users of global accounts, directories, and subaccounts.

### [Working with Environments Using the btp CLI \[page 2360\]](#)

Use the SAP BTP command line interface (btp CLI) to manage runtime environment instances in a subaccount. For example, enable the Cloud Foundry environment by creating a Cloud Foundry org (environment instance).

### [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

Use the SAP BTP command line interface (btp CLI) to manage the multitenant applications to which a subaccount is entitled to subscribe.

### [Working with External Resource Providers Using the btp CLI \[page 2362\]](#)

Use the SAP BTP command line interface (btp CLI) to get details, or to create or delete resource provider instances in a global account.

### [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)

SAP BTP supports identity federation. Its concept is to reuse the user bases of identity providers. To use a custom identity provider, your global account or subaccount in SAP BTP must have a trust relationship to the identity provider you want to use.

#### [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)

User authorizations are managed by assigning role collections to users (for example, Subaccount Administrator). Use the SAP BTP command-line interface (btp CLI) to manage roles and role collections, and to assign role collections to users.

#### [Managing Signing Keys for Access Tokens \[page 2368\]](#)

Use the SAP BTP command line interface (btp CLI) to manage signing keys for access tokens in the subaccount.

#### [Managing Security Settings \[page 2369\]](#)

Use the SAP BTP command line interface (btp CLI) to display and update the security settings for the subaccount.

#### [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)

Use the SAP BTP command line interface (btp CLI) to manage API credentials, which enable you to access the REST APIs of the SAP Authorization and Trust Management service.

#### [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

Use the SAP BTP command line interface to perform various operations related to your platforms, attached service brokers, service instances, and service bindings.

### 6.1.3.4.1 Working with Global Accounts, Directories, and Subaccounts Using the btp CLI

Use the SAP BTP command line interface (btp CLI) to manage operations with global accounts, directories, and subaccounts.

#### Working with Global Accounts

##### → Tip

These global account commands are always executed in the global account you're logged in to. You don't need to pass the global account parameter, even if you've targeted a subaccount or directory. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command...	Command help
Get details about a global account, and the account structure (directories and subaccounts) of the global account.	<code>btp get accounts/global-account</code>	<a href="#">btp get accounts/global-account</a>

Task	Run the command...	Command help
Update the display name and/or description of a global account.	btp update accounts/ global-account	<a href="#">btp update accounts/global-account</a>

For more information, see [Global Accounts \[page 94\]](#).

## Working with Directories

Directories allow you to organize and manage your subaccounts according to your technical and business needs.

### → Tip

By default, all commands are executed in the global account you're logged in to. To change the target to a directory, use `btp target -dir <my-directory-id>`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command...	Command help
Get details about a directory and list the subaccounts and subdirectories in the directory	<code>btp get accounts/directory</code>	<a href="#">btp get accounts/directory</a>
Create a directory	<code>btp create accounts/ directory</code>	<a href="#">btp create accounts/directory</a>
Update a directory	<code>btp update accounts/ directory</code>	<a href="#">btp update accounts/directory</a>
Delete a directory	<code>btp delete accounts/ directory</code>	<a href="#">btp delete accounts/directory</a>
Change the set of enabled features (user and entitlement management) for a directory	<code>btp enable accounts/ directory</code>	<a href="#">btp enable accounts/directory</a>
List the user-defined labels assigned to a directory	<code>btp list accounts/labels</code>	<a href="#">btp list accounts/label</a>

Task	Run the command...	Command help
List the custom properties assigned to a directory	btp list accounts/custom-property	<a href="#">btp list accounts/custom-property</a>

① Note

Custom properties are deprecated. Custom properties support only single values per key and are now replaced by the string array `labels`, which supports multiple values per key. Use `btp list accounts/labels` instead.

For more information, see [Directories \[page 96\]](#).

## Working with Subaccounts

→ Tip

By default, all commands are executed in the global account you're logged in to. To change the target to a subaccount, use `btp target -sa <my-subaccount-id>`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command...	Command help
List all subaccounts in a global account	btp list accounts/subaccount	<a href="#">btp list accounts/subaccount</a>
Get details about a subaccount	btp get accounts/subaccount	<a href="#">btp get accounts/subaccount</a>
Create a subaccount	btp create accounts/subaccount	<a href="#">btp create accounts/subaccount</a>
Update a subaccount	btp update accounts/subaccount	<a href="#">btp update accounts/subaccount</a>
Delete a subaccount	btp delete accounts/subaccount	<a href="#">btp delete accounts/subaccount</a>
Move a subaccount	btp move accounts/subaccount	<a href="#">btp move accounts/subaccount</a>
List the user-defined labels assigned to a subaccount	btp list accounts/labels	<a href="#">btp list accounts/label</a>

Task	Run the command...	Command help
List the custom properties assigned to a subaccount	<code>btp list accounts/custom-property</code>	<a href="#">btp list accounts/custom-property</a>
<strong> ⓘ Note</strong>  Custom properties are deprecated. Custom properties support only single values per key and are now replaced by the string array <code>labels</code> , which supports multiple values per key. Use <code>btp list accounts/labels</code> instead.		
Get all available regions for global account	<code>btp list accounts/available-region</code>	<a href="#">btp list accounts/available-region</a>

For more information, see [Subaccounts \[page 95\]](#).

**Parent topic:** Commands in the btp CLI [page 2354]

## **Related Information**

- Setting Entitlements Using the btp CLI [page 2359]
  - Working with Environments Using the btp CLI [page 2360]
  - Working with Multitenant Applications Using the btp CLI [page 2361]
  - Working with External Resource Providers Using the btp CLI [page 2362]
  - Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant [page 2363]
  - Managing Users and Their Authorizations Using the btp CLI [page 2366]
  - Managing Signing Keys for Access Tokens [page 2368]
  - Managing Security Settings [page 2369]
  - Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service [page 2370]
  - Working with Resources of the SAP Service Manager Using the btp CLI [page 2371]
  - Set a Target for Subsequent Commands with btp target [page 2346]
  - Labels [page 98]
  - btp CLI Command Reference

## 6.1.3.4.2 Setting Entitlements Using the btp CLI

Use the SAP BTP command line interface (btp CLI) to set entitlements to define the functionality or permissions available for users of global accounts, directories, and subaccounts.

### → Tip

By default, all commands are executed in the global account you're logged into. To change this to a directory or subaccount, use `btp target`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command...	Command help
Get all the entitlements and quota assignments for a global account, directories, and subaccounts	<code>btp list accounts/entitlement</code>	<a href="#">btp list accounts/entitlement</a>
Assign or update an entitlement to a subaccount or a directory	<code>btp assign accounts/entitlement</code>	<a href="#">btp assign accounts/entitlement</a>

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Working with Environments Using the btp CLI \[page 2360\]](#)

[Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)

[Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)

[Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)

[Managing Signing Keys for Access Tokens \[page 2368\]](#)

[Managing Security Settings \[page 2369\]](#)

[Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)

[Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

[btp CLI Command Reference](#)

[Entitlements and Quotas \[page 100\]](#)

## 6.1.3.4.3 Working with Environments Using the btp CLI

Use the SAP BTP command line interface (btp CLI) to manage runtime environment instances in a subaccount. For example, enable the Cloud Foundry environment by creating a Cloud Foundry org (environment instance).

### → Tip

By default, all commands are executed in the global account you're logged in to. To change this target to a subaccount, use `btp target -sa <my-subaccount-id>`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command ...	Command help
Get all available environments for a subaccount	<code>btp list accounts/available-environment</code>	<a href="#">btp list accounts/available-environment</a>
Get details about an environment available for a subaccount	<code>btp get accounts/available-environment</code>	<a href="#">btp get accounts/available-environment</a>
Get all environment instances of a subaccount	<code>btp list accounts/environment-instance</code>	<a href="#">btp list accounts/environment-instance</a>
Get a specific environment instance of a subaccount	<code>btp get accounts/environment-instance</code>	<a href="#">btp get accounts/environment-instance</a>
Create an environment instance in a subaccount	<code>btp create accounts/environment-instance</code>	<a href="#">btp create accounts/environment-instance</a>
Update the plan and/or configuration parameters of an environment in a subaccount	<code>btp update accounts/environment-instance</code>	<a href="#">btp update accounts/environment-instance</a>
Delete environment instances of a subaccount	<code>btp delete accounts/environment-instance</code>	<a href="#">btp delete accounts/environment-instance</a>

### → Tip

For information about using these btp CLI commands with the Cloud Foundry environment, see [Org Management Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2423\]](#).

For information about using btp CLI with the project "Kyma" environment, see the [Enable SAP BTP, Kyma Runtime Using the Command Line](#)  tutorial and the [Creating SAP BTP, Kyma runtime via the SAP BTP cli](#)  blog.

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Setting Entitlements Using the btp CLI \[page 2359\]](#)

[Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)  
[Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)  
[Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)  
[Managing Signing Keys for Access Tokens \[page 2368\]](#)  
[Managing Security Settings \[page 2369\]](#)  
[Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)  
[Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)  
[btp CLI Command Reference](#)

#### 6.1.3.4.4 Working with Multitenant Applications Using the btp CLI

Use the SAP BTP command line interface (btp CLI) to manage the multitenant applications to which a subaccount is entitled to subscribe.

##### → Tip

By default, all commands are executed in the global account you're logged in to. To change this target to a subaccount, use `btp target -sa <my-subaccount-id>`. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

Task	Run the command...	Command help
Get all applications to which a subaccount is entitled to subscribe	<code>btp list accounts/subscription</code>	<a href="#">btp list accounts/subscription</a>
Get details of a multitenant application in a subaccount	<code>btp get accounts/subscription</code>	<a href="#">btp get accounts/subscription</a>
Subscribe to an application from a subaccount	<code>btp subscribe accounts/subaccount</code>	<a href="#">btp subscribe accounts/subaccount</a>
Update the plan of an existing subscription	<code>btp update accounts/subscription</code>	<a href="#">btp update accounts/subscription</a>
<b>ⓘ Note</b>		
You can update a subscription plan only if additional plans for the application are entitled to the subaccount you're using and if your subscription is eligible for a plan update.		
Unsubscribe an application from a subaccount	<code>btp unsubscribe accounts/subaccount</code>	<a href="#">btp unsubscribe accounts/subaccount</a>

For more information, see [Subscribe to Multitenant Applications Using the Cockpit \[page 2198\]](#).

**Parent topic:** Commands in the btp CLI [page 2354]

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)  
[Setting Entitlements Using the btp CLI \[page 2359\]](#)  
[Working with Environments Using the btp CLI \[page 2360\]](#)  
[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)  
[Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)  
[Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)  
[Managing Signing Keys for Access Tokens \[page 2368\]](#)  
[Managing Security Settings \[page 2369\]](#)  
[Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)  
[Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)  
[btp CLI Command Reference](#)

### 6.1.3.4.5 Working with External Resource Providers Using the btp CLI

Use the SAP BTP command line interface (btp CLI) to get details, or to create or delete resource provider instances in a global account.

#### ⓘ Note

The use of this functionality is subject to the availability of the supported non-SAP cloud vendors in your country or region.

Creating a resource provider instance allows your global account to connect to your provider account on a non-SAP cloud vendor. Through this channel, you can then consume remote service resources that you already own and which are supported by SAP BTP.

Task	Run the command...	Command help
List all resource provider instances in a global account	<code>btp list accounts/resource-provider</code>	<a href="#">btp list accounts/resource-provider</a>
Get details about a resource provider instance	<code>btp get accounts/resource-provider</code>	<a href="#">btp get accounts/resource-provider</a>
Create a resource provider instance	<code>btp create accounts/resource-provider</code>	<a href="#">btp create accounts/resource-provider</a>

Task	Run the command...	Command help
Update a resource provider instance	btp update accounts/resource-provider	<a href="#">btp update accounts/resource-provider</a>
Delete a resource provider instance	btp delete accounts/resource-provider	<a href="#">btp delete accounts/resource-provider</a>

For more information, see [Managing Resource Providers \[page 2147\]](#).

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

- [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)
- [Setting Entitlements Using the btp CLI \[page 2359\]](#)
- [Working with Environments Using the btp CLI \[page 2360\]](#)
- [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
- [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)
- [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)
- [Managing Signing Keys for Access Tokens \[page 2368\]](#)
- [Managing Security Settings \[page 2369\]](#)
- [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)
- [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)
- [btp CLI Command Reference](#)

### 6.1.3.4.6 Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant

SAP BTP supports identity federation. Its concept is to reuse the user bases of identity providers. To use a custom identity provider, your global account or subaccount in SAP BTP must have a trust relationship to the identity provider you want to use.

#### Note

We recommend that you always use SAP Cloud Identity Services as a single identity provider for SAP BTP. If you use corporate identity providers, connect them to your SAP Cloud Identity Services tenant, which then acts as a hub.

## Prerequisites

- You have a tenant of SAP Cloud Identity Services.  
For more information, see [Tenant Model and Licensing](#) in the documentation for SAP Cloud Identity Services.
- The SAP Cloud Identity Services tenant is associated with the customer IDs of the relevant global account of SAP BTP.  
For more information, see [Reuse SAP Cloud Identity Services Tenants for Different Customer IDs](#) in the documentation for SAP Cloud Identity Services.
- Make sure that the email addresses of all users in your identity provider are unique and correct.

### ⓘ Note

We recommend that you request your own SAP Cloud Identity Services tenant (see [Getting a Tenant](#)).

## Managing Trust Configurations

### → Tip

All of these commands can be executed in the global account or in a subaccount. To choose the level, use the `btp target` command. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

To manage trust with your identity provider, use the following commands.

```
btp create security/trust --idp my-tenant --subaccount my-subaccount-id --name my-trust-configuration --description "My Trust Configuration"
```

See [Trust and Federation with Identity Providers \[page 2204\]](#).

Task	Run the command...	Command help
List all trust configurations that are configured for your global account or subaccount tenants to which you can connect this global account or subaccount	<code>btp list security/trust</code>	<a href="#">btp list security/trust</a>
Get details about a trust configuration	<code>btp get security/trust</code>	<a href="#">btp get security/trust</a>
Establish trust from a global account or subaccount to an SAP Cloud Identity Services tenant to which you can connect this global account or subaccount	<code>btp create security/trust</code>	<a href="#">btp create security/trust</a>
Update a trust configuration of a global account or subaccount	<code>btp update security/trust</code>	<a href="#">btp update security/trust</a>

Task	Run the command...	Command help
Delete a trust configuration from a global account or subaccount	btp delete security/trust	<a href="#">btp delete security/trust</a>
Migrate from SAML trust to OpenID Connect trust	btp migrate security/trust	<a href="#">btp migrate security/trust</a>
Roll back the migration of an existing SAML trust with any identity provider to OpenID Connect trust with an SAP Cloud Identity Services tenant	btp restore security/trust	<a href="#">btp restore security/trust</a>

## Finding Available SAP Cloud Identity Services Tenants

Task	Run the command...	Command help
List all SAP Cloud Identity Services	btp list security/available-idp	<a href="#">btp list security/available-idp</a>
Get details about an SAP Cloud Identity Services tenant that is available for a global account or a subaccount	btp get security/available-idp To see available tenants, run btp list security/available-idp	<a href="#">btp get security/available-idp</a>

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

- [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)
- [Setting Entitlements Using the btp CLI \[page 2359\]](#)
- [Working with Environments Using the btp CLI \[page 2360\]](#)
- [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
- [Working with External Resource Providers Using the btp CLI \[page 2362\]](#)
- [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)
- [Managing Signing Keys for Access Tokens \[page 2368\]](#)
- [Managing Security Settings \[page 2369\]](#)
- [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)
- [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

## 6.1.3.4.7 Managing Users and Their Authorizations Using the btp CLI

User authorizations are managed by assigning role collections to users (for example, Subaccount Administrator). Use the SAP BTP command-line interface (btp CLI) to manage roles and role collections, and to assign role collections to users.

### → Tip

All of these commands can be executed in the global account, a directory, or in a subaccount. To choose the level, use the `btp target` command. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

### Managing Users and Assigning Role Collections

Role collections are user-related authorizations that allow access to resources and services. You give users permissions by assigning role collections to them. All users in the global accounts, directories, and subaccounts are stored in identity providers, either in the default or in a custom identity provider. When the first role collection assignment to a user happens, SAP BTP creates a copy of this user in the global account, directory, or subaccount. This copy of the user is called shadow user.

When you do the first role collection assignment to a user through the btp CLI, such a shadow user is created by default. If you want to prevent this, you need to pass the `--create-user-if-missing` parameter with value `false`.

See [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#) and [User and Member Management \[page 104\]](#).

Task	Run the command ...	Command help
List users	<code>btp list security/user</code>	<a href="#">btp list security/user</a>
Get details about a specific user, including role collections	<code>btp get security/user</code>	<a href="#">btp get security/user</a>
Delete a user	<code>btp delete security/user</code>	<a href="#">btp delete security/user</a>
Assign a role collection to a user, user group, or to an attribute	<code>btp assign security/role-collection</code>	<a href="#">btp assign security/role-collection</a>
Unassign a role collection from a user, user group, or from an attribute	<code>btp unassign security/role-collection</code>	<a href="#">btp unassign security/role-collection</a>

## Managing Roles

A role is an instance of a role template; you can build a role based on a role template and assign the role to a role collection. See [Add Roles to Role Collections on the Application Level \[page 2293\]](#).

Task	Run the command ...	Command help
List apps	btp list security/app	<a href="#">btp list security/app</a>
Get details about a specific application	btp get security/app To get the ID of a specific app, run <code>btp list security/app</code>	<a href="#">btp get security/app</a>
List roles	btp list security/role	<a href="#">btp list security/role</a>
Get details about a specific role	btp get security/role	<a href="#">btp get security/role</a>
Create a role	btp create security/role	<a href="#">btp create security/role</a>
Delete a role	btp delete security/role	<a href="#">btp delete security/role</a>
Add a role to a role collection	btp add security/role	<a href="#">btp add security/role</a>
Remove a role from a role collection	btp remove security/role	<a href="#">btp remove security/role</a>

## Managing Role Collections

Role collections consist of roles, which, in turn, are based on role templates. Role collections are specific to account entities, that is, there are different role collections in global accounts, subaccounts, and directories. There are predefined role collections, such as *Global Account Administrator* and *Subaccount Viewer*. For more information, see [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#) and [Working with Role Collections \[page 2274\]](#).

Task	Run the command ...	Command help
List role collections	btp list security/role-collection	<a href="#">btp list security/role-collection</a>
Get details about a specific role collection	btp get security/role-collection	<a href="#">btp get security/role-collection</a>

Task	Run the command ...	Command help
Create a role collection	btp create security/role-collection	<a href="#">btp create security/role-collection</a>
Change the description of a role collection	btp update security/role-collection	<a href="#">btp update security/role-collection</a>
Delete a role collection	btp delete security/role-collection	<a href="#">btp delete security/role-collection</a>

**Parent topic:** Commands in the btp CLI [page 2354]

## Related Information

- [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)
- [Setting Entitlements Using the btp CLI \[page 2359\]](#)
- [Working with Environments Using the btp CLI \[page 2360\]](#)
- [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
- [Working with External Resource Providers Using the btp CLI \[page 2362\]](#)
- [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)
- [Managing Signing Keys for Access Tokens \[page 2368\]](#)
- [Managing Security Settings \[page 2369\]](#)
- [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)
- [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)
- [Security Administration: Managing Authentication and Authorization \[page 2202\]](#)
- [Set a Target for Subsequent Commands with btp target \[page 2346\]](#)
- [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#)
- [btp CLI Command Reference](#)

### 6.1.3.4.8 Managing Signing Keys for Access Tokens

Use the SAP BTP command line interface (btp CLI) to manage signing keys for access tokens in the subaccount.

You can create and manage signing keys for access tokens.

#### ⓘ Note

The newly created key only becomes active once you enable it. The number of keys is restricted to 2 per subaccount.

When a signing key is enabled, all newly requested tokens are signed with this key and the existing signing key is disabled. It's also possible to list all signing keys, no matter whether they're enabled or disabled. You can also delete a disabled signing key.

For more information, see [Rotate Signing Keys of Access Tokens](#).

Task	Run the command ...	Command help
Create a new signing key for access tokens	<code>btp create security/token-key</code>	<a href="#">btp create security/token-key</a>
Enable an existing key as signing key for access tokens	<code>btp enable security/token-key</code>	<a href="#">btp enable security/token-key</a>
Delete a disabled signing key for access tokens	<code>btp delete security/token-key</code>	<a href="#">btp delete security/token-key</a>
List the existing signing keys for access tokens and indicates which key is currently enabled	<code>btp list security/token-key</code>	<a href="#">btp list security/token-key</a>

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

- [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)
- [Setting Entitlements Using the btp CLI \[page 2359\]](#)
- [Working with Environments Using the btp CLI \[page 2360\]](#)
- [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
- [Working with External Resource Providers Using the btp CLI \[page 2362\]](#)
- [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)
- [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)
- [Managing Security Settings \[page 2369\]](#)
- [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)
- [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

### 6.1.3.4.9 Managing Security Settings

Use the SAP BTP command line interface (btp CLI) to display and update the security settings for the subaccount.

The security settings comprise configuration settings, such as information about signing keys, URLs for cross origin resource sharing or iframing, and about bindings.

For more information, see [Configure Trusted Domains for Multi-environment Subaccounts \[page 2315\]](#) and [Configure Token Policy for SAP Authorization and Trust Management Service \[page 2316\]](#).

Task	Run the command ...	Command help
Show the security settings of a subaccount	btp list security/settings	<a href="#">btp list security/settings</a>
Update the security settings of a subaccount	btp update security/settings	<a href="#">btp update security/settings</a>

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

- [Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)
- [Setting Entitlements Using the btp CLI \[page 2359\]](#)
- [Working with Environments Using the btp CLI \[page 2360\]](#)
- [Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)
- [Working with External Resource Providers Using the btp CLI \[page 2362\]](#)
- [Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)
- [Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)
- [Managing Signing Keys for Access Tokens \[page 2368\]](#)
- [Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)
- [Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

### 6.1.3.4.10 Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service

Use the SAP BTP command line interface (btp CLI) to manage API credentials, which enable you to access the REST APIs of the SAP Authorization and Trust Management service.

If you want to access the REST APIs of SAP Authorization and Trust Management service in multi-environment subaccounts, global accounts, or directories, you must provide the required API credentials. One main use case is for customers who want to provision users including their authorizations, for example into a global account or a directory. They need credentials (with a client ID) for provisioning of users, especially when using SAP Cloud Identity Services - Identity Provisioning.

The API credentials are only passed on when they are created. They can't be retrieved later. If lost, new credentials must be generated.

See [Application Security Descriptor Configuration Syntax \[page 545\]](#) for the credential types and [SAP Business Accelerator Hub](#) for the APIs of the SAP Authorization and Trust Management service.

Task	Run the command ...	Command help
Create API credentials	btp create security/api-credential	<a href="#">btp create security/api-credentials</a>
Show all API credentials	btp list security/api-credential	<a href="#">btp list security/api-credential</a>
Show details about specific API credentials	btp get security/api-credential	<a href="#">btp get security/api-credential</a>
Delete API credentials	btp delete security/api-credential	<a href="#">btp delete security/api.credentials</a>

**Parent topic:** [Commands in the btp CLI \[page 2354\]](#)

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Setting Entitlements Using the btp CLI \[page 2359\]](#)

[Working with Environments Using the btp CLI \[page 2360\]](#)

[Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)

[Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)

[Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)

[Managing Signing Keys for Access Tokens \[page 2368\]](#)

[Managing Security Settings \[page 2369\]](#)

[Working with Resources of the SAP Service Manager Using the btp CLI \[page 2371\]](#)

## 6.1.3.4.11 Working with Resources of the SAP Service Manager Using the btp CLI

Use the SAP BTP command line interface to perform various operations related to your platforms, attached service brokers, service instances, and service bindings.

You can also get information about the service plans and service offerings associated with your subaccount.

For more information about the SAP Service Manager, see [SAP Service Manager](#).

### → Tip

All of these commands are executed in subaccounts. If you know you are working in a specific subaccount, we recommend using the `btp target` command to set the target to this subaccount. Then you won't have

to provide the subaccount ID with every command call. See [Set a Target for Subsequent Commands with btp target \[page 2346\]](#).

For detailed descriptions of all SAP Service Manager CLI commands, see [SAP Service Manager Commands for SAP BTP command line interface](#).

## Managing Platforms

Task	Run the command ...	Command help
List all registered platforms in the current subaccount	btp list services/platform	<a href="#">btp list services/platform</a>
Get details about a specific platform registered in the current subaccount	btp get services/platform	<a href="#">btp get services/platform</a>
Register a new platform in the current subaccount	btp register services/platform	<a href="#">btp register services/platform</a>
Update an existing platform registered in the current subaccount	btp update services/platform	<a href="#">btp update services/platform</a>
Unregister an existing platform in the current subaccount	btp unregister services/platform	<a href="#">btp unregister services/platform</a>

## Managing Service Brokers

Task	Run the command ...	Command help
List all registered brokers in the current subaccount	btp list services/broker	<a href="#">btp list services/broker</a>
Get a specific service broker in the current subaccount	btp get services/broker	<a href="#">btp get services/broker</a>
Register a new service broker in the current subaccount	btp register services/broker	<a href="#">btp register services/broker</a>
Update an existing service broker in the current subaccount	btp update services/broker	<a href="#">btp update services/broker</a>
Unregister an existing service broker in the current subaccount	btp unregister services/broker	<a href="#">btp unregister services/broker</a>

## Managing Service Instances

Task	Run the command ...	Command help
List all service instances associated with the current subaccount.	<code>btp list services/instance</code>	<a href="#">btp list services/instance</a>
Get details about a specific service instance associated with the current subaccount.	<code>btp get services/instance</code>	<a href="#">btp get services/instance</a>
Create a new service instance of the service you want to consume.	<code>btp create services/instance</code>	<a href="#">btp create services/instance</a>
Delete an existing service instance.	<code>btp delete services/instance</code>	<a href="#">btp delete services/instance</a>
Share a service instance	<code>btp share services/instance</code>	
Unshare a service instance	<code>btp unshare services/instance</code>	

### ⓘ Note

The optional `--show-parameters` available for `btp get services/instance` and `btp get services/binding` commands returns parameters only if the service offering associated with the instance or binding for which to view parameters has the `instances_retrievable` or `bindings_retrievable` parameter values respectively set to `true`.

### → Tip

To check associated service offering's `instances_retrievable` value, run the following commands:

1. `btp get services/instance` and find the `plan_id` for the instance. Use the plan ID in
2. `btp get services/plan` and check the `offering_id` for the instance. Use the offering ID in
3. `btp get services/offering` and see whether the `instances_retrievable` parameter's value is set to `true`.

Start with `btp get services/binding` and then use the same commands for plan and offering details if you're checking the `bindings_retrievable` value.

## Managing Service Bindings

Task	Run the command ...	Command help
List all service bindings associated with the current subaccount.	btp list services/binding	<a href="#">btp list services/binding</a>
Get details about a specific service binding associated with the current subaccount.	btp get services/binding	<a href="#">btp get services/binding</a>
Create a new binding between an existing service instance and an application.	btp create services/binding	<a href="#">btp create services/binding</a>
Delete an existing binding between a service instance and an application.	btp delete services/binding	<a href="#">btp delete services/binding</a>

## Viewing Service Plans

Task	Run the command ...	Command help
List all service plans of services available for consumption that are associated with your current subaccount.	btp list services/plan	<a href="#">btp list services/plan</a>
Get details about a specific service plan of a service that is available for consumption and associated with your current subaccount.	btp get services/plan	<a href="#">btp get services/plan</a>

## Viewing Service Offerings

Task	Run the command ...	Command help
List all service offerings associated with your current subaccount.	btp list services/offering	<a href="#">btp list services/offering</a>
Get details about a specific service offering associated with your subaccount.	btp get services/offering	<a href="#">btp get services/offering</a>

**Parent topic:** Commands in the btp CLI [page 2354]

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the btp CLI \[page 2355\]](#)

[Setting Entitlements Using the btp CLI \[page 2359\]](#)

[Working with Environments Using the btp CLI \[page 2360\]](#)

[Working with Multitenant Applications Using the btp CLI \[page 2361\]](#)

[Working with External Resource Providers Using the btp CLI \[page 2362\]](#)

[Managing Trust from SAP BTP to an SAP Cloud Identity Services Tenant \[page 2363\]](#)

[Managing Users and Their Authorizations Using the btp CLI \[page 2366\]](#)

[Managing Signing Keys for Access Tokens \[page 2368\]](#)

[Managing Security Settings \[page 2369\]](#)

[Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)

[btp CLI Command Reference](#)

### 6.1.3.5 Troubleshooting and Support

Troubleshooting and support information for the btp CLI.

#### Make sure that you are using a recent version of the btp CLI client

We recommend checking for updates of the client on a regular basis. Most new functionality is added via the CLI server, so no update from your side is required. But if new functionality is added to the client, the only way to get it is by downloading and installing a newer version.

Run the command `btp` or `btp --info` to find out which version of the client you are using. Then check on our [download page](#) for the current version and install the latest client.

Use `btp --help` to display an overview of all commands. If new commands are available but cannot be used with your client version, these commands are marked with a notice to update your client. If you run such a command, you get an error message pointing you to the client version that you need.

#### I cannot log on

To log on, we recommend to run `btp login --sso` or `btp login` without parameters. Then confirm the server URL that is proposed by the CLI and provide your credentials when you are prompted. After successful authentication, the btp CLI will log you into your global account, or, if you have more than one, will show all available global accounts for you to select the one you need.

To log on with SAP Universal ID, you need to use the `--sso` parameter. Otherwise log on with the password associated with your account (S-user or P-user) in the default identity provider, SAP ID service. If you've forgotten this password and this user is associated with your SAP Universal ID user, reset your password.

For more information, see SAP Note [3085908](#) and [Log in with Single Sign-On \[page 2341\]](#).

## I'd like to setup a global account with the command line

If you don't know how to get started, you might want to have a look at the appropriate workflow: [Setting Up a Trial Account From the Command Line \[page 154\]](#) or [Setting Up a Global Account via the Command Line \[page 162\]](#).

## I'd like to see a more detailed output (verbose mode)

If you get an error message that is not helpful enough, or if you want to find out more about what exactly happened after you ran a command, use the `btp --verbose` option with your command and run it again. The `btp` CLI will try to execute the command again, but also print information about each step of the command execution. This might help you understand what exactly went wrong and provide hints for you to solve the issue. If you need to create a support request, providing this information helps us analyze the error and provide a solution more quickly.

Here is an example of the `btp list security/role-collection` command call without the option:

### ↔ Sample Code

```
btp list security/role-collection
name                           description
Global Account Administrator   Administrative access to the global account
Global Account Viewer         Read-only access to the global account
OK
```

If you use `btp --verbose list security/role-collection`, the output is much more lengthy. It contains information such as client version and the current context, the correlation ID, and request and response details. This output may include sensitive information, so we recommend to use the verbose mode only for troubleshooting, not as default, and only on your local machine.

### ⚠ Caution

The `--verbose` option prints input data that you pass as parameters through files. For example, if you pass a `.json` file as a parameter without the `--verbose` option, its content is sent directly to the backend service to be processed. If you use the `--verbose` option, the `btp` CLI prints the content of the file to the terminal. If there is sensitive information in such a file, the `btp` CLI cannot filter it out, as it doesn't understand its semantics.

Use the verbose output only if you can control the output of your terminal, i.e. only locally. It should not be used in production or shared environments, where logs may be persisted or even transported out of the system.

## ↔ Sample Code

```
btp --verbose list security/role-collection
2019/12/18 10:35:21.309235 main.go:34: Support trace activated.
2019/12/18 10:35:21.309284 main.go:35: Client version 1.x.x running with
command line: --verbose list security/role-collection
2019/12/18 10:35:21.309349 clifiles.go:65: Config file path was set to /
Users/my-user/Library/Caches/.btp/config.json
2019/12/18 10:35:21.309386 login.go:138: Reading configuration from &{/Users/my-user/Library/Caches/.btp config.json commands.json}
2019/12/18 10:35:21.309578 main.go:60: CLI server URL: http://localhost:8080
2019/12/18 10:35:21.309591 main.go:60: Global account subdomain: my-ga-
subdomain
2019/12/18 10:35:21.309594 main.go:60: Subaccount: not set
2019/12/18 10:35:21.309597 main.go:60: Configuration file: /Users/my-
user/Library/Caches/.btp/config.json
2019/12/18 10:35:21.309603 login.go:138: Reading configuration from &{/Users/my-user/Library/Caches/.btp config.json commands.json}
2019/12/18 10:35:21.309663 main.go:68: Successfully loaded login config.
2019/12/18 10:35:21.311758 protocol.go:41: Loading command metadata from
http://localhost:8080/commandMetadata/v-1
2019/12/18 10:35:21.311860 protocol.go:248: Adding correlation ID
1234abcde-ab12-12ab-34cd-123456abcdef to request http://localhost:8080/
commandMetadata/v-1
2019/12/18 10:35:21.319441 protocol.go:53: Received response 200 from request
http://localhost:8080/commandMetadata/v-1
2019/12/18 10:35:21.322441 protocol.go:161: Executing command list security/
role-collection with parameters map[subaccount:{0 }]
2019/12/18 10:35:21.322622 protocol.go:205: Sending request:
{POST http://localhost:8080/command/v-1/security/role-collection?list
HTTP/1.1 1 1 map[Content-Type:[application/json;charset=UTF-8] X-Cpcli-
Subdomain:[my-ga-subdomain]] {"paramValues":{"subaccount":null}}} 0x12c6380
35 [] false localhost:8080 map[] map[] <nil> map[] <nil> <nil> <nil>
0xc0000ae070}
to http://localhost:8080/command/v-1/security/role-collection?list
2019/12/18 10:35:21.322642 protocol.go:248: Adding correlation ID 1234abcde-
ab12-12ab-34cd-123456abcdef to request http://localhost:8080/command/v-1/
security/role-collection?list
2019/12/18 10:35:24.683689 protocol.go:214: Received response 200 from
request http://localhost:8080/command/v-1/security/role-collection?list
2019/12/18 10:35:24.690916 main.go:196: Getting response mapping for command
result code 200
2019/12/18 10:35:24.690943 main.go:198: Response mapping: {[200] ok {table
[name description]}}
name description
Global Account Administrator Administrative access to the global account
Global Account Viewer Read-only access to the global account
OK
```

## I need support

Use component BC-CP-TOOLS-CLI to contact support.

If a correlation ID is printed with your error message, please provide it with your support ticket. The correlation ID is created for each command execution, and is passed with all of its steps. It allows the identification of all log messages related to a command execution.

See [Getting Support \[page 3148\]](#).

## 6.1.4 Account Administration Using APIs

SAP BTP provides REST APIs that enable you to perform administrative tasks on the global account, directory, and subaccount level, such as creating or updating subaccounts, monitoring usage information, managing access, and managing service resources.

### Related Information

[Account Administration Using APIs of the SAP Cloud Management Service \[page 2378\]](#)

[Monitoring Usage Information Using APIs of the SAP Usage Data Management Service \[page 2409\]](#)

[Accessing Administration Using APIs of the SAP Authorization and Trust Management Service \[page 2416\]](#)

[Managing Service Resources Using the APIs of the SAP Service Manager \[page 2420\]](#)

### 6.1.4.1 Account Administration Using APIs of the SAP Cloud Management Service

Provides information about using the APIs of the SAP Cloud Management service for SAP BTP (technical name: cis) to manage some of the administrative operations in your accounts.

The APIs of the core services for SAP BTP allow you to manage, build, and extend the core capabilities of SAP BTP. Each core service focuses on a different aspect of the platform, such as the management of accounts and product entitlements, and the registration and provisioning of subscriptions for multitenant applications and services.

The APIs of the SAP Cloud Management service are included in the core services for SAP BTP.

The following table provides an overview of the SAP Cloud Management service APIs:

API / Microservice	Description	Region Availability <sup>(1)</sup>
Accounts	Provides functions to manage the directories and subaccounts in your global accounts' structure in SAP BTP.	Central region <sup>(2)</sup>
Entitlements	Provides functions to manage product entitlements and assignments across your global account, directories, and subaccounts.	Central region <sup>(2)</sup>
Provisioning	Provides functions to manage the provisioning of your environment instances, multitenant application subscriptions, and services for subaccounts in their corresponding region.	All regions
Events	Provides functions to get information about events relating to administrative operations in your accounts.	All regions

(1) For more information about available regions, see [Regions \[page 17\]](#).

(2) Refers to the eu10 Europe (Frankfurt) region, unless your global account is located in the China (Shanghai) and Government Cloud (US) regions then use the landscape domain of your central region.

For detailed information about the APIs, including their specifications and endpoints, go to [SAP Core Services for SAP BTP](#) on SAP Business Accelerator Hub

Alternatively, go to `https://events-service.<app domain>.<landscape domain>/swagger-index.html` where you can try out the APIs.

### ⌚ Example

If your global account is running on the US East (VA) region, use this URL:

```
https://events-service.cfapps.us10.hana.ondemand.com/swagger-index.html
```

### → Remember

- When using the Accounts and Entitlements APIs, make sure the endpoint you are using points to the landscape domain of the central region.
- To call the core platform API methods, you must obtain an access token. See [Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#).

For information about the APIs of the SAP Service Manager service, see [Working with Service Management APIs](#).

For additional information about using the SAP Cloud Management service APIs with subaccounts in the Neo environment, see [Working with Cloud Management Tools Feature Set B in the Neo Environment](#).

## Related Information

[Error Response Format \[page 2406\]](#)

[Asynchronous Jobs \[page 2408\]](#)

[Using the Events Service APIs \[page 2388\]](#)

[Rate Limiting \[page 2408\]](#)

[Monitoring Usage Information Using APIs of the SAP Usage Data Management Service \[page 2409\]](#)

[Using SAP SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 379\]](#)

[Using Platform APIs](#)

[Account Model \[page 94\]](#)

[Environments \[page 47\]](#)

[Entitlements and Quotas \[page 100\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Data Protection and Privacy \[page 3109\]](#)

## 6.1.4.1.1 Getting an Access Token for SAP Cloud Management Service APIs

The APIs of the SAP Cloud Management service for SAP BTP are protected with the OAuth 2.0 Password grant type and, in some cases, also the Client Credentials grant type. This procedure guides you through the steps to create an OAuth client and obtain an access token from SAP Authorization and Trust Management service (xsuaa) to call the APIs of the SAP Cloud Management service.

### ⓘ Note

The Client Credentials grant type is currently available for the SAP Cloud Management service only when creating the instances of this service on a subaccount level by using the SAP Service Manager API, CLI, or when creating an instance of an SAP Cloud Management service by using the SAP BTP cockpit and selecting the *Other* environment.

For more information, see [Consuming Services in Other Environments Using the Service Management Instances](#).

The scopes included in the access token depend on the service plan you chose for the SAP Cloud Management service. For the list of the available service plans, see [SAP Cloud Management Service - Service Plans \[page 2385\]](#).

## Prerequisites

Your global account admin has entitled at least one of the plans of the SAP Cloud Management service in your subaccount. See [SAP Cloud Management Service - Service Plans \[page 2385\]](#) and [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

## Procedure

1. Create a service instance for the SAP Cloud Management service (`cis`).

When you create the service instance, specify the following:

- The name of the service plan you want to use. See [SAP Cloud Management Service - Service Plans \[page 2385\]](#).
- A name for the new service instance.
- (Optional) Using the `directoryId` parameter, you can specify the ID of a directory in the account hierarchy for which you want to allow directory administrators to perform account management actions, such as creating subaccounts and setting entitlements in the directory, using the APIs of the [Accounts and Entitlements services](#). The specified directory must exist in the current global account and the `AUTHORIZATIONS` feature must be enabled for the directory. You can apply this parameter only with the central plan of the SAP Cloud Management service (`cis`).

There are several options available to create instances depending on the environment you use.

- To create a service instance in the Cloud Foundry environment using the SAP BTP cockpit or the cf CLI, see [Creating Service Instances \[page 250\]](#).

- To create a service instance in other environments using the Service Manager Control (SMCTL) CLI or SAP Service Manager APIs, see [Consuming Services in Other Environments Using the Service Management Instances](#).

You can also create and manage service instances and bindings in other environments by using the following APIs in the Accounts service:

URL	Name
<code>POST/accounts/v1/subaccounts/{subaccountGUID}/serviceManagementBinding</code>	Create a Service Manager binding
<code>GET/accounts/v1/subaccounts/{subaccountGUID}/serviceManagementBinding</code>	Get a Service Manager binding
<code>DELETE/accounts/v1/subaccounts/{subaccountGUID}/serviceManagementBinding</code>	Delete a Service Manager binding

For more information about the APIs, see the Subaccount Operations section of the Accounts service in the [SAP Business Accelerator Hub](#).

#### ① Note

If you use the SAP Service Manager API, CLI, or the SAP BTP cockpit to create the service instance of the SAP Cloud Management service (`cis`), you can get a Client Credentials grant type token by specifying the following parameter during the instance creation: `{"grantType": "clientCredentials"}`.

If you don't specify this parameter, the Password grant type is chosen by default.

#### → Remember

For `central-viewer` or `local-viewer` service plans, you can only use the Client Credentials grant type.

- To create a service instance in Kyma using the Kyma dashboard, see [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#).

#### → Recommendation

If you aren't working in Cloud Foundry, Kyma, or Kubernetes, use the SAP Service Manager to create and manage service instances. These instances are platform agnostic and can be deployed and integrated with any other environment of your choice.

## 2. Create a service key.

When you create the service key, specify:

- The name of the service instance for which to create the service key.
- A name for the service key.

There are several options available to create service keys depending on the environment you use.

- To create a service key in the Cloud Foundry environment using the SAP BTP cockpit or the cf CLI, see [Creating Service Keys \[page 255\]](#).

- To create bindings in other environments, see [Consuming Services in Other Environments Using the Service Management Instances](#).
- To create credentials for calling the service and retrieving information in the Kyma environment, see [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#).

### ↔ Sample Code

The following example shows the service key information displayed in the Cloud Foundry environment:

```
{
  "endpoints": {
    "external_provider_registry_url": <external provider registry URL>,
    "accounts_service_url": <accounts service URL>,
    "entitlements_service_url": <entitlements service URL>,
    "events_service_url": <events service URL>,
    "order_processing_url": <order processing service URL>,
    "provisioning_service_url": <provisioning service URL>,
    "saas_registry_service_url": <saas registry service URL>,
  },
  "grant_type": "user_token",
  "uaa": {
    ...
    "clientid": <client_id>,
    "clientsecret": <client_secret>,
    "url": <uaa_url>,
    ...
  }
}
```

3. Use the `uaa_url`, `clientid`, and `clientsecret` to request an access token using the following commands:

### ↔ Sample Code

For Windows OS (Password grant type):

```
curl -L -X POST "<uaa_url>/oauth/token" ^
-H "Content-Type: application/x-www-form-urlencoded" ^
-u "<clientid>:<clientsecret>" ^
-d "grant_type=password" ^
-d "username=<user_email>" ^
-d "password=<password>"
```

### ↔ Sample Code

For Windows OS (Client Credentials grant type):

```
curl -L -X POST "<uaa_url>/oauth/token" ^
-H "Content-Type: application/x-www-form-urlencoded" ^
-u "<clientid>:<clientsecret>" ^
-d "grant_type=client_credentials" ^
```

### ↔ Sample Code

For Mac OS (Password grant type):

```
curl -L -X POST '<uaa_url>/oauth/token' \
-H 'Content-Type: application/x-www-form-urlencoded' \
-u '<clientid>:<clientsecret>' \
-d 'grant_type=password' \
```

```
-d 'username=<user_email>' \
-d 'password=<password>'
```

## ↔ Sample Code

For Mac OS (Client Credentials grant type):

```
curl -L -X POST '<uaa_url>/oauth/token' \
-H 'Content-Type: application/x-www-form-urlencoded' \
-u '<clientid>:<clientsecret>' \
-d 'grant_type=client_credentials' \
```

## ⓘ Note

If two-factor authentication (2FA) is activated on the SAP BTP landscape, then you need to append the passcode generated by the SAP Authenticator to your password.

For example, if your password is `Abcd` and the authenticator-generated passcode is `1234`, enter the password as `Abcd1234`.

Two-factor authentication is only relevant for Password grant type authorization.

## ⓘ Note

The access token received also contains the scopes that are granted for this access token. Therefore, only APIs that require one of these scopes can be used with this access token.

```
{
  "access_token": "<access_token>",
  "token_type": "bearer",
  "expires_in": 43199,
  "scope": "<xsappname>.job.read <xsappname>.event.read"
}
```

See [Get Access to the APIs Using the apiaccess Service Plan](#)

4. Call the SAP Cloud Management service APIs in SAP API Business Accelerator Hub with the `access_token` that you received in the previous step:
  1. Choose one of the endpoints of your instance. See step 2 and the sample code for a service instance in Cloud Foundry.
  2. Browse to `<endpoint url>/swagger-ui.html`.
  3. Enter the `api_key` and choose [Authorize](#).

## ⓘ Note

Use the `<access_token>` that you received in the step 3 to compose the `api_key` in the format:  
`bearer <access_token>`

5. Choose the API that you want to call, provide the relevant parameters, and choose [Try it out!](#)

## Trying Out SAP Cloud Management Service API in SAP Business Accelerator Hub

To learn more about SAP Business Accelerator Hub, see [What Is the SAP Business Accelerator Hub?](#)

### Note

In the SAP Business Accelerator Hub, you find SAP Cloud Management Service APIs under the **Core Services for SAP BTP** API Package.

To start working with SAP Cloud Management Service APIs in SAP Business Accelerator Hub, you'll need to configure an environment. See [Trying Out APIs in a Configured Environment](#).

### Note

You configure an environment by populating all mandatory fields in the [Create New Environment](#) wizard.

Refer to steps 2 and 3 of this topic to see how to get the values for Client ID, Client Secret, and Token URL fields.

## 6.1.4.1.1 SAP Cloud Management Service - Service Plans

Describes the service plans for the SAP Cloud Management service for SAP BTP and the scopes that they provide. The technical name for this service is `cis`.

Service Display Name	Technical Name	Service Plan	Scopes	Additional Configuration Parameters
Cloud Management Service	<code>cis</code>	central:Service plan for using SAP Cloud Management service APIs to manage your global accounts, subaccounts, directories, and entitlements.	<ul style="list-style-type: none"><li>• <code>global-ac-count.read</code></li><li>• <code>global-ac-count.update</code></li><li>• <code>global-ac-count.subac-count.read</code></li><li>• <code>global-ac-count.subac-count.create</code></li><li>• <code>global-ac-count.subac-count.update</code></li><li>• <code>global-ac-count.subac-count.delete</code></li><li>• <code>global-ac-count.account-directory.read</code></li><li>• <code>global-ac-count.account-directory.create</code></li><li>• <code>global-ac-count.account-directory.update</code></li><li>• <code>global-ac-count.account-directory.delete</code></li><li>• <code>global-ac-count.entitlement</code></li><li>• <code>global-ac-count.entitlement.subac-count.update</code></li><li>• <code>global-account.region.read</code></li><li>• <code>catalog.product.update</code></li><li>• <code>catalog.product.delete</code></li><li>• <code>directory.entitlement.update</code></li></ul>	<ul style="list-style-type: none"><li>• <code>grantType</code>: Choose whether to get a Client Credentials or Password grant type token when using the SAP Service Manager API, CLI, or the SAP BTP cockpit to create the service instance of the SAP Cloud Management service (<code>cis</code>). If you don't specify this parameter, the Password grant type is chosen by default.</li><li>• <code>directoryId</code>: ID of a directory to allow directory administrators to use the APIs of the <a href="#">Accounts and Entitlements services</a> to perform account management actions, such as creating subaccounts and setting entitlements, in the directory.</li></ul> <p>For more information, see <a href="#">Getting an Access Token for SAP Cloud Management Service APIs [page 2380]</a>.</p>

Service Display Name	Technical Name	Service Plan	Scopes	Additional Configuration Parameters
Cloud Management Service		local:Service plan for using SAP Cloud Management service APIs to manage your environments and subscriptions to multitenant applications.	<ul style="list-style-type: none"> <li>• directory.entitlement.read</li> <li>• job.read</li> <li>• cis-central.event.read</li> <li>• account-automation-request.read</li> </ul>	<p>grantType: Choose whether to get a Client Credentials or Password grant type token when using the SAP Service Manager API, CLI, or the SAP BTP cockpit to create the service instance of the SAP Cloud Management service (<i>cis</i>). If you don't specify this parameter, the Password grant type is chosen by default. For more information, see <a href="#">Getting an Access Token for SAP Cloud Management Service APIs [page 2380]</a>.</p>

Service Display Name	Technical Name	Service Plan	Scopes	Additional Configuration Parameters
Cloud Management Service		central-viewer:Service plan for using SAP Cloud Management service APIs to read your global accounts, subaccounts, directories, and service entitlements and assignments.	<ul style="list-style-type: none"> <li>global-account.read</li> <li>global-account.subaccount.read</li> <li>global-account.account-directory.read</li> <li>global-account.entitlement.read</li> <li>global-account.region.read</li> <li>directory.entitlement.read</li> <li>job.read</li> <li>automation-module.read</li> <li>automation-solution.read</li> <li>automation-request.read</li> <li>account-automation-request.read</li> <li>account-automation-solution.read</li> </ul>	grantType: Only the Client Credentials grant type is available when using the SAP Cloud Management service APIs to read the SAP BTP entities included with this plan. For more information, see <a href="#">Getting an Access Token for SAP Cloud Management Service APIs [page 2380]</a> .
Cloud Management Service		local-viewer:Service plan for using SAP Cloud Management service APIs to read your environments and subscriptions to SaaS multitenant applications.	<ul style="list-style-type: none"> <li>subaccount.entitlement.read</li> <li>subaccount.environment.read</li> <li>subaccount.application.subscription.read</li> <li>event.read</li> <li>job.read</li> </ul>	grantType: Only the Client Credentials grant type is available when using the SAP Cloud Management service APIs to read the SAP BTP entities included with this plan. For more information, see <a href="#">Getting an Access Token for SAP Cloud Management Service APIs [page 2380]</a> .
SaaS Provisioning Service	saas-registry	application: Service plan for application owners to manage the lifecycle of multitenant applications with SAP Software-as-a-Service Provisioning service APIs.	<ul style="list-style-type: none"> <li>job.read</li> <li>subscription.read</li> <li>subscription.write</li> <li>entitlement.read</li> </ul>	See configuration JSON file properties in <a href="#">Register the Multitenant Application to the SAP SaaS Provisioning Service [page 370]</a> .

For information about assigning these plans to a subaccount, see [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

## Related Information

[Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#)

### 6.1.4.1.2 Using the Events Service APIs

The Events service provides REST APIs that collect information about events relating to account administrative operations in the microservices of the SAP Cloud Management service for SAP BTP, such as Accounts, Entitlements, Provisioning, and the SAP SaaS Provisioning service, within central and local regions.

You can filter the events by various query parameters, such as a specific time frame, event type, or the type of entity associated with the event.

You can also use the APIs to get all available event types.

Here are some examples of scenarios that can be implemented or built by reacting to account administrative events:

- A global account admin can implement an automatic tagging system for newly created subaccounts by polling for subaccount creation events, `Subaccount_Creation`, from the Accounts service and then adding custom property to each one of the subaccounts according to predefined criteria.
- Set up automatic notifications when a global account admin assigns quota to subaccounts. This scenario uses the `SubaccountEntitlements_Update` event from the Entitlements service.
- Send automatic alerts when a directory is deleted. This scenario uses the event `AccountDirectory_Deletion` from the Accounts service.
- Use the subscription events information, `SubaccountAppSubscription_Creation`, from the SAP SaaS Provisioning service, to send emails notifying when applications are subscribed to from a subaccount.

**Permissions:** A service instance of the SAP Cloud Management service is required. The scopes required to get the events are available for all of the plans for the SAP Cloud Management service. See [SAP Cloud Management Service - Service Plans \[page 2385\]](#).

For more information about permissions, see [Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#).

**Base URI:** `https://events-service.<app domain>.<landscape domain>`

## Methods

HTTP Method	Action	URI
GET	Get events	events-service.<app domain>.<landscape domain>/cloud-management/v1/events
GET	Get event types	events-service.<app domain>.<landscape domain>/cloud-management/v1/events/types

## Related Information

[Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#)

### 6.1.4.1.2.1 Get Events

Get all events associated with administrative operations in your accounts.

The events you get depend on the scopes you used to access the API.

To learn more about the scopes, see [SAP Cloud Management Service - Service Plans \[page 2385\]](#).

## Prerequisites

You have obtained an access token with the \$XSAPPNAME.event.read scope.

## Request

**URI:** `https://events-service.<app domain>.<landscape domain>/cloud-management/v1/events`

**HTTP Method:** `GET`

## Query Parameters

Parameter Name	Required	Parameter Type	Description
entityId	No	Array of strings	The ID of the entity associated with the event.
entityType	No	Array of strings	The type of entity associated with the event.  For example Subaccount, Directory, or Tenant.

Parameter Name	Required	Parameter Type	Description
eventType	No	Array of strings	<p>The type of the event that was triggered.</p> <p>There are two groups of event types, Central Events and Local Events group.</p> <p>The group you get depends on the scopes granted to you after you authorized to use the API.</p> <p>You can query any event listed in the group that is relevant for your scope.</p> <p>To retrieve all available event types, use the Event Types API.</p> <p>See <a href="#">Get Event Types [page 2401]</a>.</p> <p>The examples of some of the events for both groups:</p> <ul style="list-style-type: none"> <li>• <b>Central Events group:</b> GlobalAccount_Update, AccountDirectory_Creation, AccountDirectory_Update, AccountDirectory_Update_Type, AccountDirectory_Deletion, Subaccount_Creation, Subaccount_Deletion, Subaccount_Update, Subaccount_Move, AccountDirectoryTenant_Creation, AccountDirectoryTenant_Deletion, GlobalAccountEntitlements_Update, EntityEntitlements_Update, EntityEntitlements_Move</li> <li>• <b>Local Events group:</b> SubaccountAppSubscription_Cre</li> </ul>

Parameter Name	Required	Parameter Type	Description
			ation, SubaccountAppSubscription_Deletion, SubaccountAppSubscription_Update, AppRegistration_Creation, AppRegistration_Deletion, AppRegistration_Update, SubaccountTenant_Creation, SubaccountTenant_Update, SubaccountTenant_Deletion, EnvironmentInstance_Creation, EnvironmentInstance_Deletion, EnvironmentInstances_Deletion
fromActionTime	No	Integer	<p>Start date and time to query the events by the action that triggered them.</p> <p>Use the Unix epoch time in milliseconds (you can find an online converter from a regular date-time format to the Unix epoch time format).</p>

#### ❖ Example

Monday, June 1, 2020 9:40:22 AM is 1590993622000 in Unix epoch milliseconds time.

Parameter Name	Required	Parameter Type	Description
fromCreationTime	No	Integer	Start date and time to query the events by when they were created.  Use the Unix epoch time in milliseconds (you can find an online converter from a regular date-time format to the Unix epoch time format).
			<p> <b>Example</b></p> <p>Monday, June 10, 2020 04:32:22 AM is 1591752742000 in Unix epoch milliseconds time.</p>
<b>id</b>	No	Array of integers	The ID of the event.
<b>pageNum</b>	No	Integer	The page number to retrieve.
<b>pageSize</b>	No	Integer	The number of events to retrieve per page (max = 150).
<b>searchParams</b>	No	JSON Object	Additional search parameters that depend on the type of the events.
<b>sortField</b>	No	String	Field by which to sort the events.
<b>sortOrder</b>	No	String	Sort order for the events.  Can be ascending or descending.  Available values: ASC,DESC.

Parameter Name	Required	Parameter Type	Description
toActionTime	No	Integer	<p>End date and time to query the events by the action that triggered them.</p> <p>Use the Unix epoch time in milliseconds (you can find an online converter from a regular date-time format to the Unix epoch time format).</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <span style="color: #ccc;">✖</span> <b>Example</b>            Monday, June 4, 2020 11:40:22 AM is 1591260022000 in Unix epoch milliseconds time.         </div>
toCreationTime	No	Integer	<p>End date and time to query the events by when they were created.</p> <p>Use the Unix epoch time in milliseconds (you can find an online converter from a regular date-time format to the Unix epoch time format).</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <span style="color: #ccc;">✖</span> <b>Example</b>            Monday, June 6, 2020 12:32:22 AM is 1591392742000 in Unix epoch milliseconds time.         </div>

## Response

Returns the list of events, and its associated objects.

**Content Type:** [JSON](#)

## Response Status and Error Codes

Code	Description
200	Found events (OK)
400	<p>Bad Request</p> <p>Possible reasons:</p> <ul style="list-style-type: none"><li>• The requested event type doesn't belong to the events group associated with the scope you used to access the API</li><li>• Requested event type is unknown</li><li>• Query arguments are invalid</li></ul>
401	<p>Unauthorized</p> <p>Possible reasons:</p> <ul style="list-style-type: none"><li>• Invalid token</li><li>• A token with the scopes for central region was used to access the Events APIs in a local region.</li></ul>
403	Forbidden
404	Not Found
429	Rate Limit Exceeded
500	<p>Internal Server Error</p> <p>Possible reasons:</p> <ul style="list-style-type: none"><li>• Failed to obtain global account or subaccount info</li></ul>

## Response Objects and Their Parameters

### BusinessEventsResponseCollection

A collection of the event objects associated with the API call and the used scopes.

Parameter Name	Parameter Type	Description
events	Array of JSON objects:  BusinessEventResponseObject	List of the event objects.  See the next table  BusinessEventResponseObject for its parameters.
morePages	Boolean	Whether there are more pages with event objects to return.
pageNum	Integer	The current page number.
total	Integer	Total number of results.
totalPages	Integer	Total number of pages.

### BusinessEventResponseObject

Includes details about an event in the list.

Parameter Name	Parameter Type	Description
actionTime	String	The time the action triggered the event. The format is Unix epoch time in milliseconds.
creationTime	String	The time the event record was created. The format is Unix epoch time in milliseconds.
details	JSON object	Contains description and details about the requested events.
entityId	String	The ID of the entity associated with the event.
entityType	String	The type of the entity associated with the event.
eventOrigin	String	The service that reported the event.

Parameter Name	Parameter Type	Description
eventType	String	<p>The type of the event that was triggered.</p> <p>There are two groups of event types: Local Events and Central Events group.</p> <p>Only event types that belong to one of the groups is returned as the result of a single API call.</p> <p>The event types group you get depends on the scope you used to access the API.</p> <p>The examples of some of the events for both groups:</p> <ul style="list-style-type: none"> <li>• Central Events group: GlobalAccount_Update, AccountDirectory_Creation, AccountDirectory_Update, AccountDirectory_Update_Type, AccountDirectory_Deletion, Subaccount_Creation, Subaccount_Deletion, Subaccount_Update, Subaccount_Move, AccountDirectory_Tenant_Creation, AccountDirectory_Tenant_Deletion, GlobalAccountEntitlements_Update, EntityEntitlements_Update, EntityEntitlements_Move</li> <li>• Local Events group: SubaccountAppSubscription_Creation, SubaccountAppSubscription_Deletion, SubaccountAppSubscription_Update, AppRegistration_Creation, AppRegistration_Deletion, AppRegistration_Update, SubaccountTenant_Creation, SubaccountTenant_Update, SubaccountTenant_Deletion, EnvironmentInstance_Creation, EnvironmentInstance_Deletion, EnvironmentInstances_Deletion</li> </ul>
globalAccountGUID	String	The unique ID of the global account associated with the event.
id	Integer	The ID of the event.

## Response Example

```
{  
    "total": 3,  
    "totalPages": 1,  
    "pageNum": 0,  
    "morePages": false,  
    "events": [  
        {  
            "id": 584299,  
            "actionTime": 1593494674668,  
            "creationTime": 1593494674709,  
            "details": {  
                "description": "Subaccount created.",  
                "guid": "6699b187-d17c-4783-bc8c-3263690d4aac",  
                "parentGuid": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",  
                "displayName": "trial",  
                "subaccountDescription": null,  
                "region": "us10-staging",  
                "jobLocation": null,  
                "subdomain": "6a193bbetrial",  
                "betaEnabled": false,  
                "expiryDate": null  
            },  
            "globalAccountGUID": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",  
            "entityId": "6699b187-d17c-4783-bc8c-3263690d4aac",  
            "entityType": "Subaccount",  
            "eventOrigin": "accounts-service",  
            "eventType": "Subaccount_Creation"  
        },  
        {  
            "id": 584300,  
            "actionTime": 1593494675583,  
            "creationTime": 1593494676435,  
            "details": {  
                "description": "Addition and update of entitlements in a global account",  
                "entitlementItems": {  
                    "entitlements": [  
                        {  
                            "productId": "trial",  
                            "name": "TRIAL",  
                            "amount": 1,  
                            "effectiveDate": null,  
                            "allowedSubaccountIDs": null  
                        }  
                    ],  
                    "origin": null  
                }  
            },  
            "globalAccountGUID": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",  
            "entityId": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",  
            "entityType": "GlobalAccount",  
            "eventOrigin": "entitlements-service",  
            "eventType": "GlobalAccountEntitlements_Update"  
        },  
        {  
            "id": 584315,  
            "actionTime": 1593494726523,  
            "creationTime": 1593494727151,  
            "details": {  
                "description": "Assignment of entitlements from global account to subaccount",  
                "servicePlanAssignments": "{\"destinationlite\":{\"decreased\":{},\"increased\":{\"6699b187-d17c-4783-bc8c-3263690d4aac\":{\"amount\":10,\"servicePlanAssignmentId\":21589}},\"serviceName\":\"destination\",\"servicePlanName\":\"lite\",\"servicePlanUniqueIdentifier\":\"destinationlite\",\"isEntitlementUnlimited\":false,\"assignmentLevel\":\"SUBACCOUNT\"},\"mongodb\"}  
            }  
        }  
    ]  
}
```

```

edium\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21590 }, \\"serviceName\" :\"mongodb\", \"s
ervicePlanName\" :\"medium\", \"servicePlanUniqueIdentifier\" :\"mongodbmedium\", \"i
sEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"applicationru
ntime\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :100, \"servicePlanAssignmentId\" :21591 }, \\"serviceName\" :\"APPLICATION
_RUNTIME\", \\"servicePlanName\" :\"MEMORY\", \\"servicePlanUniqueIdentifier\" :\"appli
cationruntime\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT
\"}, \\"cis-local\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :5, \"servicePlanAssignmentId\" :21592 }, \\"serviceName\" :\"cis\", \\"servi
cePlanName\" :\"local\", \\"servicePlanUniqueIdentifier\" :\"cis-
local\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"ht
ml5-apps-repo-app-host\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :10, \"servicePlanAssignmentId\" :21593 }, \\"serviceName\" :\"html5-apps-
repo\", \\"servicePlanName\" :\"app-host\", \\"servicePlanUniqueIdentifier\" :\"html5-
apps-repo-app-
host\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"rab
bitmqsmall\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21594 }, \\"serviceName\" :\"rabbitmq\", \\"s
ervicePlanName\" :\"small\", \\"servicePlanUniqueIdentifier\" :\"rabbitmqsmall\", \"i
sEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"rabbitmq-
virtualhost\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :5, \"servicePlanAssignmentId\" :21595 }, \\"serviceName\" :\"rabbitmq\", \\"s
ervicePlanName\" :\"virtualhost\", \\"servicePlanUniqueIdentifier\" :\"rabbitmq-
virtualhost\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT
\"}, \\"postgresqlmedium\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21596 }, \\"serviceName\" :\"postgresql\",
\"servicePlanName\" :\"medium\", \\"servicePlanUniqueIdentifier\" :\"postgresqlmedium
\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"postgre
sqllarge\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21597 }, \\"serviceName\" :\"postgresql\",
\"servicePlanName\" :\"large\", \\"servicePlanUniqueIdentifier\" :\"postgresqllarge\"
, \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"screening
hits-application\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21598 }, \\"serviceName\" :\"screeninghits
\", \\"servicePlanName\" :\"saas-
application\", \\"servicePlanUniqueIdentifier\" :\"screeninghits-
application\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT
\"}, \\"redis-dev\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :10, \"servicePlanAssignmentId\" :21599 }, \\"serviceName\" :\"redis\", \\"se
rvicePlanName\" :\"dev\", \\"servicePlanUniqueIdentifier\" :\"redis-
dev\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"redi
slarge\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21600 }, \\"serviceName\" :\"redis\", \\"ser
vicePlanName\" :\"large\", \\"servicePlanUniqueIdentifier\" :\"redislarge\", \"isEntit
lementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"rabbitmq-dev\" :
{ \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-bc8c-3263690d4aac\" :
{ \"amount\" :10, \"servicePlanAssignmentId\" :21601 }, \\"serviceName\" :\"rabbitmq\", \\"s
ervicePlanName\" :\"dev\", \\"servicePlanUniqueIdentifier\" :\"rabbitmq-
dev\", \"isEntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"mong
odblarge\" : { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-
bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21602 }, \\"serviceName\" :\"mongodb\", \\"s
ervicePlanName\" :\"large\", \\"servicePlanUniqueIdentifier\" :\"mongodblarge\", \"isE
ntitlementUnlimited\" :false, \"assignmentLevel\" :\"SUBACCOUNT\" }, \\"rabbitmqlarge\"
: { \"decreased\" : {}, \"increased\" : { \\"6699b187-d17c-4783-bc8c-3263690d4aac\" :
{ \"amount\" :1, \"servicePlanAssignmentId\" :21603 }, \\"serviceName\" :\"rabbitmq\", \\"

```

```

servicePlanName\" : \"large\", \"servicePlanUniqueIdentifier\" : \"rabbitmqlarge\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"featureFlagslite\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 10, \"servicePlanAssignmentId\" : 21604}}, \"serviceName\" : \"featureflagslite\", \"servicePlanName\" : \"lite\", \"servicePlanUniqueIdentifier\" : \"featureflagslite\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"auditlog-viewer_standard\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 10, \"servicePlanAssignmentId\" : 21605}}, \"serviceName\" : \"auditlog-viewer\", \"servicePlanName\" : \"standard\", \"servicePlanUniqueIdentifier\" : \"auditlog-viewer_standard\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"redismedium\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21606}}, \"serviceName\" : \"redis\", \"servicePlanName\" : \"medium\", \"servicePlanUniqueIdentifier\" : \"redismedium\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"rabbitmqmedium\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21607}}, \"serviceName\" : \"rabbitmq\", \"servicePlanName\" : \"medium\", \"servicePlanUniqueIdentifier\" : \"rabbitmqmedium\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"postgresqlsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 2147483647, \"servicePlanAssignmentId\" : 21608}}, \"serviceName\" : \"postgresql\", \"servicePlanName\" : \"small\", \"servicePlanUniqueIdentifier\" : \"postgresqlsmall\", \"isEntitlementUnlimited\" : true, \"assignmentLevel\" : \"SUBACCOUNT\"}, \"mongodb_xsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21609}}, \"serviceName\" : \"mongodb\", \"servicePlanName\" : \"xsmall\", \"servicePlanUniqueIdentifier\" : \"mongodb_xsmall\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"connectivitylite\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 10, \"servicePlanAssignmentId\" : 21610}}, \"serviceName\" : \"connectivity\", \"servicePlanName\" : \"lite\", \"servicePlanUniqueIdentifier\" : \"connectivitylite\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"rabbitmqxsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 2147483647, \"servicePlanAssignmentId\" : 21611}}, \"serviceName\" : \"rabbitmq\", \"servicePlanName\" : \"xsmall\", \"servicePlanUniqueIdentifier\" : \"rabbitmqxsmall\", \"isEntitlementUnlimited\" : true, \"assignmentLevel\" : \"SUBACCOUNT\", \"portalServicessite\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 10, \"servicePlanAssignmentId\" : 21612}}, \"serviceName\" : \"portal-services\", \"servicePlanName\" : \"site\", \"servicePlanUniqueIdentifier\" : \"portalservicessite\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\"}, \"redisxsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21613}}, \"serviceName\" : \"redis\", \"servicePlanName\" : \"xsmall\", \"servicePlanUniqueIdentifier\" : \"redisxsmall\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"mongodbsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21614}}, \"serviceName\" : \"mongodb\", \"servicePlanName\" : \"small\", \"servicePlanUniqueIdentifier\" : \"mongodbsmall\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"redissmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21615}}, \"serviceName\" : \"redis\", \"servicePlanName\" : \"small\", \"servicePlanUniqueIdentifier\" : \"redissmall\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\", \"postgreqlxsmall\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21616}}, \"serviceName\" : \"postgresql\", \"servicePlanName\" : \"xsmall\", \"servicePlanUniqueIdentifier\" : \"postgreqlxsmall\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\"}, \"mongodb-dev-large\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" : {\"amount\" : 1, \"servicePlanAssignmentId\" : 21617}}, \"serviceName\" : \"mongodb\", \"servicePlanName\" : \"dev-large\", \"servicePlanUniqueIdentifier\" : \"mongodb-dev-large\", \"isEntitlementUnlimited\" : false, \"assignmentLevel\" : \"SUBACCOUNT\"}, \"screening-application\" : {\"decreased\" : {}, \"increased\" : {\"6699b187-d17c-4783-bc8c-3263690d4aac\" :

```

```
{
  "amount": 1, "servicePlanAssignmentId": 21618}, "serviceName": "screening", "servicePlanName": "saas-application", "servicePlanUniqueIdentifier": "screening-application", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 150, "servicePlanAssignmentId": 21619}, "serviceName": "postgresql", "servicePlanName": "dev", "servicePlanUniqueIdentifier": "postgresql-dev", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 1, "servicePlanAssignmentId": 21620}, "serviceName": "sample-saas-app-i337243", "servicePlanName": "test", "servicePlanUniqueIdentifier": "sample-saas-app-i337243-test", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 5, "servicePlanAssignmentId": 21621}, "serviceName": "jobscheduler", "servicePlanName": "standard", "servicePlanUniqueIdentifier": "jobscheduler-standard", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 5, "servicePlanAssignmentId": 21622}, "serviceName": "cis", "servicePlanName": "central", "servicePlanUniqueIdentifier": "cis-central", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}],
  "subaccountIdToRegion": {"amount": 1, "servicePlanAssignmentId": 21618, "serviceName": "screening", "servicePlanName": "saas-application", "servicePlanUniqueIdentifier": "screening-application", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 150, "servicePlanAssignmentId": 21619, "serviceName": "postgresql", "servicePlanName": "dev", "servicePlanUniqueIdentifier": "postgresql-dev", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 1, "servicePlanAssignmentId": 21620, "serviceName": "sample-saas-app-i337243", "servicePlanName": "test", "servicePlanUniqueIdentifier": "sample-saas-app-i337243-test", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 5, "servicePlanAssignmentId": 21621, "serviceName": "jobscheduler", "servicePlanName": "standard", "servicePlanUniqueIdentifier": "jobscheduler-standard", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}, {"amount": 5, "servicePlanAssignmentId": 21622, "serviceName": "cis", "servicePlanName": "central", "servicePlanUniqueIdentifier": "cis-central", "isEntitlementUnlimited": false, "assignmentLevel": "SUBACCOUNT"}},
  "globalAccountGUID": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",
  "entityId": "a7410cf4-84a5-46ac-9c15-52e34d283eb2",
  "entityType": "GlobalAccount",
  "eventOrigin": "entitlements-service",
  "eventType": "SubaccountEntitlements_Update"
}
]
}
}
```

## Related Information

[Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#)

[Using the Events Service APIs \[page 2388\]](#)

[Get Event Types \[page 2401\]](#)

### 6.1.4.1.2.2 Get Event Types

Get all available event types, including their categories and their available search parameters.

The event types you get are either for a central or for a local region, and the region you get depends on the scopes you used to access the API.

To learn more about the scopes, see [SAP Cloud Management Service - Service Plans \[page 2385\]](#).

## Prerequisites

You have obtained an access token with the `$XSAPPNAME.event.read` scope.

## Request

**URI:** `https://events-service.<app domain>.<landscape domain>/cloud-management/v1/events/types`

**HTTP Method:** `GET`

## Response

Returns the event types, their descriptions and categories.

**Content Type:** `JSON`

### Response Status and Error Codes

Code	Description
200	OK
401	Unauthorized Possible reasons: <ul style="list-style-type: none"><li>• Invalid token</li></ul>
403	Forbidden
404	Not Found
429	Rate Limit Exceeded
500	Internal Server Error Possible reasons: <ul style="list-style-type: none"><li>• Failed to obtain global account or subaccount info</li></ul>

### Response Objects and Their Parameters

`BusinessEventTypeResponseObject`

A JSON object that contains details about event types.

Parameter Name	Parameter Type	Description
category	String	<p>Category to which the event type belongs.</p> <p>Possible values:</p> <ul style="list-style-type: none"> <li>• LOCAL: The event is associated with the local region within a multi-region universe.</li> <li>• CENTRAL: The event is associated with the central region within a multi-region universe.</li> </ul>
description	String	The description of the event type.
searchParams	Array	<p>List of all the search parameters for the event type.</p> <p>Provided inline.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <span style="color: #800000;">❖ Example</span>  <pre>&lt;URL HOST&gt;?searchParam- Key1=searchParamValue1</pre> </div>

Parameter Name	Parameter Type	Description
type	String	<p>The type of the event that was triggered.</p> <p>There are two groups of event types: Local Events and Central Events group.</p> <p>Only event types that belong to one of the groups are returned as the result of a single API call.</p> <p>The event types group you get depends on the scope you used to access the API.</p> <p>The examples of some of the events for both groups:</p> <ul style="list-style-type: none"> <li>• Central Events group: GlobalAccount_Update, AccountDirectory_Creation, AccountDirectory_Update, AccountDirectory_Update_Type, AccountDirectory_Deletion, Subaccount_Creation, Subaccount_Deletion, Subaccount_Update, Subaccount_Move, AccountDirectory_Tenant_Creation, AccountDirectory_Tenant_Deletion, GlobalAccountEntitlements_Update, EntityEntitlements_Update, EntityEntitlements_Move</li> <li>• Local Events group: SubaccountAppSubscription_Creation, SubaccountAppSubscription_Deletion, SubaccountAppSubscription_Update, AppRegistration_Creation, AppRegistration_Deletion, AppRegistration_Update, SubaccountTenant_Creation, SubaccountTenant_Update, SubaccountTenant_Deletion, EnvironmentInstance_Creation, EnvironmentInstance_Deletion, EnvironmentInstances_Deletion</li> </ul>

## Response Example

```
{
  "SubaccountAppSubscription_Update": {
    "searchParams": [
      ...
    ]
  }
}
```

```

        "saasApplicationId",
        "saasApplicationName",
        "consumerTenantId"
    ],
    "description": "A subaccount's subscription to a multitenant application, including its dependencies, was updated.",
    "category": "LOCAL",
    "type": "SubaccountAppSubscription_Update"
},
"SubaccountAppSubscription_Creation": {
    "searchParams": [
        "saasApplicationId",
        "saasApplicationName",
        "consumerTenantId"
    ],
    "description": "A subaccount was subscribed to a multitenant application and its dependencies.",
    "category": "LOCAL",
    "type": "SubaccountAppSubscription_Creation"
},
"AppRegistration_Creation": {
    "searchParams": [
        "saasApplicationId",
        "saasApplicationName"
    ],
    "description": "A multitenant application was registered in the SaaS registry",
    "category": "LOCAL",
    "type": "AppRegistration_Creation"
},
"SubaccountTenant_Creation": {
    "searchParams": [],
    "description": "The tenant of a subaccount was created.",
    "category": "LOCAL",
    "type": "SubaccountTenant_Creation"
},
"EnvironmentInstance_Creation": {
    "searchParams": [],
    "description": "An environment instance was created. For example, a Cloud Foundry org was enabled for a subaccount.",
    "category": "LOCAL",
    "type": "EnvironmentInstance_Creation"
},
"AppRegistration_Update": {
    "searchParams": [
        "saasApplicationId",
        "saasApplicationName"
    ],
    "description": "The registration of a multitenant application in the SaaS registry was changed.",
    "category": "LOCAL",
    "type": "AppRegistration_Update"
},
"AppRegistration_Deletion": {
    "searchParams": [
        "saasApplicationId",
        "saasApplicationName"
    ],
    "description": "A multitenant application was unregistered from the SaaS registry.",
    "category": "LOCAL",
    "type": "AppRegistration_Deletion"
},
"EnvironmentInstances_Deletion": {
    "searchParams": [],
    "description": "All environment instances in a subaccount were deleted.",
    "category": "LOCAL",
    "type": "EnvironmentInstances_Deletion"
}

```

```

} ,
"SubaccountAppSubscription_Deletion": {
    "searchParams": [
        "saasApplicationId",
        "saasApplicationName",
        "consumerTenantId"
    ],
    "description": "The subscription of a subaccount to a multitenant application including its dependencies was cancelled.",
    "category": "LOCAL",
    "type": "SubaccountAppSubscription_Deletion"
},
"SubaccountTenant_Deletion": {
    "searchParams": [],
    "description": "The tenant of a subaccount was deleted.",
    "category": "LOCAL",
    "type": "SubaccountTenant_Deletion"
},
"SubaccountTenant_Update": {
    "searchParams": [],
    "description": "The tenant of a subaccount was changed.",
    "category": "LOCAL",
    "type": "SubaccountTenant_Update"
},
"EnvironmentInstance_Deletion": {
    "searchParams": [
        "environmentType",
        "tenantId",
        "subaccountGuid",
        "platformId"
    ],
    "description": "An environment instance was deleted. For example, a Cloud Foundry org, including its contents, was removed from a subaccount.",
    "category": "LOCAL",
    "type": "EnvironmentInstance_Deletion"
}
}
}

```

## Related Information

[Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#)

[Using the Events Service APIs \[page 2388\]](#)

[Get Events \[page 2389\]](#)

### 6.1.4.1.3 Error Response Format

Describes the response format for the errors from the SAP Cloud Management service for SAP BTP.

Standard HTTP codes are used to indicate successful execution or if errors occur. The response includes additional information about the error, including an application error code and human readable error description. This example represents a common error response resulting from a failed API request.

```

HTTP/1.1 400 Bad Request
{
    "error": {
        "code": 11000,

```

```

"message": "Request payload is invalid",
"target": "/accounts/v1/subaccounts/125e2778-e457-4c24-9db8-e3e10a3a0ed5",
"details": [
  {
    "code": 11001,
    "message": "description: length must be between 0 and 255"
  }
]
}

```

Parameter	Description	Type
error.code	A unique application error code. Common values are listed below.	Number
error.message	A description of the error.	String
error.target	The resource URL of the request.	String
error.details	An array of additional error codes and descriptions with further details about the error and its root cause. If this field exists, the array will contain at least one element.	Array

An application error code is not necessarily coupled with a specific HTTP status code.

## Error Codes

Provides a list of error codes and the details of the errors.

Error Code	Description
10XXX	Server errors
11XXXX	General request errors
12XXXX	General operations failures
2XXXXX	Accounts service errors
3XXXXX	Entitlements service errors
40XXXX	Tenants operations failures
41XXXX	Environment instances operations failures
42XXXX	Provisioning service errors
5XXXXX	Service Management service proxy errors
6XXXXX	Order Processing service errors
7XXXXX	CLI backend errors
8XXXXX	Asynchronous jobs failures
9XXXXX	SaaS Provisioning service errors
10XXXX	Events service errors

## 6.1.4.1.4 Asynchronous Jobs

Describes the asynchronous calls that are supported by some functionality from the SAP Cloud Management service for SAP BTP.

When the API responds with the status code [202](#), the request can be processed as an asynchronous job.

The status of the request can be fetched using the path provided in the [Location](#) header. The response of the API provided in the [Location](#) header is normally:

```
{  
  "status": "<IN_PROGRESS/COMPLETED/FAILED>",  
  "description": "<meaningful description in case one needed>"  
}
```

### ⓘ Note

The response includes the `status` (always) and `description` (optional) fields.

For example, you can trigger a brand new job.

## 6.1.4.1.5 Rate Limiting

Describes how all API requests to the SAP Cloud Management service for SAP BTP adhere to rate limiting rules.

Every API endpoint defines its own custom rate limit rule for the number of requests per time window and per identified caller.

Authenticated requests are associated either with the authenticated username, tenant ID or with the OAuth client ID. Unauthenticated requests are associated with the originating IP address, and not the user making requests.

When the rate limit is exceeded, the client receives the HTTP [429 Too Many Requests](#) response status code.

## Response Example

For example, if you try to perform more than five GET requests to the `/accounts/v1/globalAccounts` endpoint in the `accounts-service` in less than a three second time window, you will receive the following error:

```
HTTP/1.1 429 Too Many Requests  
{  
  "error": {  
    "code": 11006,  
    "message": "Request rate limit exceeded.",  
    "target": "/accounts/v1/globalAccounts",  
    "details": [  
      {  
        "code": 11007,  
        "message": "Maximum request rate limit is <X> requests in <Y> seconds"  
    ]  
  }  
}
```

```
        }
    ]
}
```

#### ⓘ Note

- The error code for the rate limit exceeded error is always [11006](#) and the details contains the list of exceeded rate limit rules with the [11007](#) error code.
- No HTTP headers are returned to show your current rate limit status (for example, the number of requests remaining in the current rate limit window).

### 6.1.4.2 Monitoring Usage Information Using APIs of the SAP Usage Data Management Service

Provides information about using the [Resource Consumption](#) APIs of the SAP Usage Data Management service for SAP BTP for gathering, storing, and making usage information available for all services and applications in all regions in a cloud deployment. This information is for the purpose of central analysis, reporting, and license auditing.

The service accumulates the information and provides reports in SAP BTP cockpit via APIs for resource planning and cross-billing purposes.

#### ⓘ Note

The SAP Usage Data Management service is not sold to SAP BTP customers as a service; however, customers can see their usage data in the cockpit, for example, in the Usage Analytics pages.

### Accessing the API

For more information about the APIs, including their specifications and endpoints, go to [Resource Consumption](#) for the list of all SAP Usage Data Management service APIs in the SAP Business Accelerator Hub.

#### → Remember

To call the core platform API methods, you must obtain an access token. See [Configuration for Using the Resource Consumption APIs in the SAP Business Accelerator Hub \[page 2410\]](#).

**Base URI:** <https://uas-reporting.cfapps.<landscape domain>/reports/v1>

Your global account admin has entitled the service plan for the SAP Usage Data Management service in your subaccount. See [SAP Usage Data Management Service - Service Plans \[page 2413\]](#).

## Methods

HTTP Method	Action	URI
GET	Get the cloud credit data history for a global account.	/cloudCreditDetails
GET	Get monthly usage reporting data for a directory and a specified time period.	/monthlyDirectoryUsage
GET	Get monthly usage-reporting data for a global account and a specified time period.	/monthlyUsage
GET	Get usage-reporting data for a subaccount.	/subaccountUsage
GET	Get monthly cost reporting data for all subaccounts for the Cloud Platform Enterprise Agreement (CPEA) consumption-based commercial model.	/monthlySubaccountsCost

## Related Information

[Configuration for Using the Resource Consumption APIs in the SAP Business Accelerator Hub \[page 2410\]](#)

[SAP Usage Data Management Service - Service Plans \[page 2413\]](#)

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

### 6.1.4.2.1 Configuration for Using the Resource Consumption APIs in the SAP Business Accelerator Hub

The **Resource Consumption** APIs of the SAP Usage Data Management service for SAP BTP are protected with OAuth 2.0 Client Credentials grant type and in some cases, also the Password grant type.

#### ⓘ Note

OAuth 2.0 Password grant type is only mandatory for the `subaccountUsage` API.

## Prerequisites

- To use the *Resource Consumption* APIs of the SAP Usage Data Management service
  - You must have service access to the relevant **uas** service plan. For information about available service plans, see [SAP Usage Data Management Service - Service Plans \[page 2413\]](#).

- When using **Password grant type** the user must be assigned one of the following role collections:

API	Role Collection Required
monthlyUsage, /monthlySubaccountsCost, and subaccountUsage	Global Account Administrator and Global Account Viewer

For more information, see [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

### ⓘ Note

To create a service instance in other environments using the Service Manager Control (SMCTL) CLI, SAP Service Manager APIs, or the SAP BTP cockpit, see [Consuming Services in Other Environments Using the SAP Service Manager Instances](#).

## Procedure

- Create a service instance for the service plans that correspond to the APIs that you want to use.  
When you create the service instance, specify the following parameters:
  - PLAN**: The name of the service plan that you want to use. See [SAP Usage Data Management Service - Service Plans \[page 2413\]](#).
  - SERVICE\_INSTANCE**: Name of the service instance.
 There are several options available to create instances depending on the environment you use.
  - To create a service instance in other environments using the Service Manager Control (SMCTL) CLI or SAP Service Manager APIs, see [Consuming Services in Other Environments Using the SAP Service Manager Instances](#).
  - To create a service instance in the Cloud Foundry environment using the SAP BTP cockpit or the cf CLI, see [Creating Service Instances \[page 250\]](#).
  - To create a service instance in Kyma using the Kyma dashboard, see [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#).

### → Recommendation

If you are not working in Cloud Foundry, Kyma, or Kubernetes, use the SAP Service Manager to create and manage service instances. These instances are platform-agnostic and can be deployed and integrated with any other environment of your choice.

- Get the OAuth 2.0 client information and uas service endpoints by creating a service key or binding. There are several options available to create service keys or bindings depending on the environment you use.
  - To create bindings in other environments, see [Consuming Services in Other Environments Using the SAP Service Manager Instances](#).
  - To create a service key in the Cloud Foundry environment using the SAP BTP cockpit or the cf CLI, see [Creating Service Keys \[page 255\]](#).
  - To create credentials for calling the service and retrieving information in the Kyma environment, see [Using SAP BTP Services in the Kyma Environment \[page 1975\]](#).

3. Use `uaa_url`, `clientid`, and `clientsecret` to request an access token using the following commands:

#### ↔ Sample Code

For Windows OS (Password grant type):

```
curl -L -X POST "<uaa_url>/oauth/token" ^
-H "Content-Type: application/x-www-form-urlencoded" ^
-u "<clientid>:<clientsecret>" ^
-d "grant_type=password" ^
-d "username=<user_email>" ^
-d "password=<password>"
```

#### ↔ Sample Code

For Windows OS (Client Credentials grant type):

```
curl -L -X POST "<uaa_url>/oauth/token" ^
-H "Content-Type: application/x-www-form-urlencoded" ^
-u "<clientid>:<clientsecret>" ^
-d "grant_type=client_credentials" ^
```

#### ↔ Sample Code

For Mac OS (Password grant type):

```
curl -L -X POST '<uaa_url>/oauth/token' \
-H 'Content-Type: application/x-www-form-urlencoded' \
-u '<clientid>:<clientsecret>' \
-d 'grant_type=password' \
-d 'username=<user_email>' \
-d 'password=<password>'
```

#### ↔ Sample Code

For Mac OS (Client Credentials grant type):

```
curl -L -X POST '<uaa_url>/oauth/token' \
-H 'Content-Type: application/x-www-form-urlencoded' \
-u '<clientid>:<clientsecret>' \
-d 'grant_type=client_credentials' \
```

#### ⓘ Note

The access token that you receive also contains the scopes that are granted for this access token. Therefore, only APIs that require one of these scopes can be used with this access token.

```
{
  "access_token": "<access_token>",
  "token_type": "bearer",
  "expires_in": 43199,
  "scope": "<xsappname>.UAS.reporting.GA_Admin"
}
```

4. Using the `<access_token>` that you received in the previous step you can call the [Resource Consumption APIs](#) using bearer authentication or you can try them directly in the SAP Business Accelerator Hub using the following steps:

1. Got to [Resource Consumption APIs](#) in the SAP Business Accelerator Hub.
  2. Choose *Select Environment* and select the *Add New Environment* option
  3. In the *Authentication Type* field, select *OAuth 2.0 Application Flow*.
  4. Provide the parameters that you have retrieved in the following fields: `clientid`, `clientsecret`, and `identityzone`.
  5. Save your changes.
5. Choose the API that you want to call, provide the relevant parameters, and choose *Run*.

To learn more about the SAP Business Accelerator Hub, see [What Is the SAP API Business Hub?](#)

## 6.1.4.2.2 Rate Limiting Rules

API requests to the SAP Usage Data Management service for SAP BTP are subject to rate limits.

Rate limits restrict the number of requests made by a caller within a defined time period. For authenticated requests, callers are identified by the authenticated user name, the tenant ID or the OAuth client ID. For non-authenticated requests, callers are identified by the originating IP address.

When you exceed the rate limit, you receive the HTTP response status code [429 Too Many Requests](#).

The rate limits are:

- The rate limit per authentication token or IP is 120 requests per second.
- The maximum payload body size per request is 80 MB.

### ❖ Example

For example, if you try to perform more than 120 GET requests to the `/reports/v1/monthlyUsage` endpoint in less than 1 second, you receive the following error:

```
<html>
<head><title>429 Too Many Requests</title></head>
<body>
<center><h1>429 Too Many Requests</h1></center>
<hr><center>openresty</center>
</body>
</html>
```

## 6.1.4.2.3 SAP Usage Data Management Service - Service Plans

Describes the plans available for the SAP Usage Data Management service for SAP BTP.

### ⓘ Note

The SAP Usage Data Management service APIs can be found in the [SAP Business Accelerator Hub](#).

Display Name	Technical Service Name	Service Plans	Scopes
Usage Data Management Service (UDM)	uas	<p>reporting-ga-admin: Service plan for using the SAP Usage Data Management service APIs.</p> <p>This microservice is used to generate reports based on resource and cost consump- tion of a global account.</p>	Reporting.GA_Admin

Display Name	Technical Service Name	Service Plans	Scopes
		<p>reporting-directory: Service plan for using the SAP Usage Data Management service APIs.</p> <p>This microservice is used to generate reports based on resource and cost consump- tion of a directory.</p> <p>When you create a service in- stance of this plan, you have to pass the directory ID in the following format:</p> <ul style="list-style-type: none"> <li>• { "directoryGuid"   : "f4c920c1-   efb7-4bda-8d0e-8   1616ab9a574" } - if     you create the service     instance using the SAP     BTP cockpit</li> <li>• -c   " { \"directoryGui   d\": \"f4c920c1-   efb7-4bda-8d0e-8   1616ab9a574\" } " -     if you create the service     instance using the cf CLI</li> </ul>	\$XSAPPNAME.reporting.Directory_Admin

**⚠ Restriction**

The directory must be located under `globalAccount` and contain the subaccount from which you are creating the service instance.

User management must be enabled for the directory, see [Manage Users in Directories \[page 2169\]](#).

## Related Information

[Configuration for Using the Resource Consumption APIs in the SAP Business Accelerator Hub \[page 2410\]](#)

### 6.1.4.3 Accessing Administration Using APIs of the SAP Authorization and Trust Management Service

The REST services of the SAP Authorization and Trust Management service (XSUAA) provide APIs that enable you to manage entities, such as roles, shadow users, and access tokens in global accounts, directories, and multi-environment subaccounts.

#### ⓘ Note

To manage Neo subaccounts, use the Neo APIs. For more information, see [Using Platform APIs](#).

## API Overview

The following table provides an overview of the APIs. For more information about the APIs and their API endpoints, see *SAP Business Accelerator Hub* in the related links.

APIs of SAP Authorization and Trust Management Service

API	Description
User Management (System for Cross-domain Identity Management (SCIM))	Provides functions to administrate the SAP Authorization and Trust Management service of the Cloud Foundry environment. Provision users from identity providers and manage roles and role collections. Use this API to manage shadow users; users the service provisions from your identity provider to the subaccount.  See <a href="#">SAP BTP XS Advanced UAA (Cloud Foundry)</a> for more information.
Authorization	Provides functions to administrate the SAP Authorization and Trust Management service of Cloud Foundry environment. Manage service instances of the SAP Authorization and Trust Management service. You can also manage roles, role templates, and role collections of your subaccount.
Identity Provider Management	Provides functions to manage identity providers in the Cloud Foundry environment.
Security Settings	Provides functions to manage the security settings of an account in Cloud Foundry environment. Use this API to manage access token validity and signing keys.

## Related Information

[APIs of SAP Authorization and Trust Management Service service on SAP Business Accelerator Hub](#)

[Using Platform APIs](#)

[Rate Limiting \[page 3083\]](#)

[Limits for Technical Artifacts of the SAP Authorization and Trust Management Service \[page 3088\]](#)

### 6.1.4.3.1 Get Access to the APIs

To enable programmatic access to the SAP Authorization and Trust Management service (XSUAA) in your global account, directory, or multi-environment subaccount, you need API credentials.

#### Context

To get access to the APIs of the SAP Authorization and Trust Management service, use the SAP BTP command-line interface (btp CLI) to create API credentials. You can get API credentials that apply for the global account, directory, or subaccount.

For more information, see [security/api-credential](#).

#### Procedure

1. Log on to the SAP BTP CLI.
2. Enter the following command in the relevant global account, subaccount, or directory:

```
btp create security/api-credential --name my-credential
```

You get API credentials, which enable you to access the [REST APIs of the SAP Authorization and Trust Management service on Business Accelerator Hub](#).

The btp CLI shows the API credential as follows:

#### ↔ Output Code

```
Name: my-credential
Credential Type: binding-secret
Read-only: false
Client Secret: <secret>
Certificate: <null>
Client ID: sb-full-access!a76125
```

With the client ID and client secret, or certificate, you can request an access token for the APIs in the targeted global account, directory, or subaccount.

## Related Information

[Managing API Credentials for Calling REST APIs of SAP Authorization and Trust Management Service \[page 2370\]](#)

[btp CLI Command Reference](#)

### 6.1.4.3.2 Call an API

Use the different endpoints of the SAP Authorization and Trust Management service APIs to manage users roles and other authorization configurations.

#### Context

Actually, you need two separate URLs. First, you need the token URL to fetch an access token. Next, you can use the API URL to call an API.

To call an API of the SAP Authorization and Trust Management service, you need an access token. See the following example how to retrieve an access token and how to call an endpoint with a cURL request.

On the *SAP Business Accelerator Hub*, we provide a sandbox system for you to make example calls to the different APIs and their endpoints.

Use the following API URLs:

`https://api.authentication.<region>.hana.ondemand.com`

#### ⓘ Note

To find the region, use a btp CLI command, for example `btp get accounts/subaccount subaccount-id`.

#### Procedure

- Configure or program your application to request a token from the OAuth authorization server.

In your call to the OAuth server, you send the client ID and client secret to the token URL appended with the /oauth/token endpoint. To illustrate the call, we use cURL in the following example.

#### ↔ Sample Code

```
curl --request POST \
  --url 'https://my-subdomain.authentication.eu10.hana.ondemand.com/oauth/
token' \
  --header 'Accept: application/json' \
  --data 'client_id=sb-full-access!a76125' \
  --data 'client_secret=<secret>' \
```

```
--data 'grant_type=client_credentials' \
--data 'response_type=token'
```

The authorization server returns a token along with other related information.

#### ↔ Output Code

```
{"access_token": "eyJhbGciOiJSUzI1N...",
"token_type": "bearer",
"expires_in": 43199,
"scope": "xs_user.write uaa.resource xs_authorization.read
          xs_idp.write xs_user.read xs_idp.read xs_authorization.write",
"jti": "be340353ac694b4cb504c6823f938647"}
```

2. To retrieve the token, use the following URL:

<https://<subdomain>.authentication.<region>.hana.ondemand.com>

#### ⓘ Note

To find the subdomain of the global account, directory, and subaccount, go to the SAP BTP cockpit and display the overview in the [Account Explorer](#).

When you have a token, you can use this token to call the APIs until its validity expires. In this case, simply get a new access token.

3. Use the value of the `access_token` property to make calls to the various API endpoints.
4. Call the endpoint of an API.

#### ↔ Sample Code

In this example, we request the list of roles of from the authorization API. With the authorization header, use the access token you received in the prerequisites after the key word `bearer`.

```
curl --request GET \
      --url https://api.authentication.eu10.hana.ondemand.com/sap/rest/
      authorization/v2/roles \
      --header 'Accept: application/json' \
      --header 'Authorization: bearer eyJhbGciOiJSUzI1N...'
```

## Results

The API returns the list of roles.

#### ↔ Output Code

```
[

  {
    "roleTemplateName": "Viewer",
    "roleTemplateAppId": "myapp!t1111",
    "name": "Viewer",
    "attributeList": [],
    "roleCapabilityIDList": [],
    "description": "Default instance",
```

```
        "scopes": [
            {
                "description": "Display forecast",
                "name": "myapp!t1111.Tourist",
                "custom-granted-apps": [],
                "granted-apps": [],
                "grant-as-authority-to-apps": [],
                "custom-grant-as-authority-to-apps": []
            }
        ]
    }
```

## Related Information

[Authorization and Trust Management APIs on the SAP Business Accelerator Hub](#)

[Implementing Custom Token Retrieval from SAP Authorization and Trust Management Service with mTLS \[page 525\]](#)

### 6.1.4.4 Managing Service Resources Using the APIs of the SAP Service Manager

Use the APIs of the SAP Service Manager to work with environments, service brokers, service instances, service bindings, service plans, and service offerings.

SAP Service Manager is the central registry for service brokers and platforms in SAP BTP.

The service allows you to consume platform capabilities in any connected runtime environment, track the creation and managing of service instances, and share services and service instances between different environments.

For the list of the available APIs, see [Working with SAP Service Manager APIs](#).

### 6.1.5 Account Administration Using Infrastructure as Code

Infrastructure as Code (IaC) automates infrastructure provisioning and management using code. It helps ensure consistency, scalability, versioning, collaboration, and documentation. The Terraform Provider for SAP BTP allows you to provision and manage SAP BTP resources as code, with open-source support.

#### What is Infrastructure as Code?

Infrastructure as Code (IaC) is an approach to managing and provisioning infrastructure resources as code, using machine-readable configuration files or scripts, instead of manually configuring them. Infrastructure

configurations are written in a programming language and usually stored in version control systems, so they can easily be shared, reviewed, and modified, enabling collaboration and reproducibility.

The following are some benefits of the IaC approach:

- **Automation:** It allows for the automation of infrastructure provisioning and management processes. Infrastructure can be created, modified, and destroyed programmatically, reducing the need for manual intervention and minimizing human error.
- **Consistency:** By defining infrastructure as code, you can ensure that your infrastructure is consistently provisioned and configured across different environments, such as development, testing, and production. This helps in reducing configuration drift and improves reliability.
- **Scalability:** IaC enables the ability to scale infrastructure resources up or down based on demand. By defining infrastructure configurations in code, you can easily replicate and deploy infrastructure resources in a consistent and repeatable manner.
- **Versioning and Collaboration:** IaC configurations can be stored in version control systems, allowing you to track changes over time and roll back to previous versions if needed. It also enables collaboration among team members, as multiple people can work on the same infrastructure codebase simultaneously.
- **Documentation:** IaC serves as a form of documentation, providing a clear and concise representation of the desired infrastructure state. It helps in understanding the infrastructure architecture and facilitates knowledge sharing within the team.

Overall, Infrastructure as Code brings the principles and practices of software development to infrastructure management, with the goal of enabling organizations to achieve greater agility, scalability, and reliability in their infrastructure operations.

## Infrastructure as Code with the Terraform Provider for SAP BTP

There is an open-source solution for using IaC for SAP BTP: The **Terraform Provider for SAP BTP**. It focuses on administration capabilities within SAP BTP global accounts. For example, it lets you manage subaccounts and directories, entitlements, environment instances, subscriptions, service instances and bindings, users and role collections, and trust configurations to custom identity providers.

### Where to Find the Terraform Provider for SAP BTP

- You find the Terraform Provider for SAP BTP in Hashicorp's Terraform registry, the central provider repository: <https://registry.terraform.io/providers/SAP/btp>.
- You find the source code and documentation in this GitHub repository: <https://github.com/SAP/terraform-provider-btp>.

### Support

The Terraform Provider for SAP BTP was built by SAP as an open-source project under the [Apache 2.0 license](#). Note that this means that there is no official SAP support for the provider. Feedback, issues, pull requests, and feature requests are handled through the [open repository on github.com](#).

### Documentation

For a complete overview of the scope of the Terraform Provider for SAP BTP, see the documentation in Hashicorp's Terraform Registry: <https://registry.terraform.io/providers/SAP/btp/latest/docs>

To learn how to get started, see the tutorial: [Get Started with the Terraform Provider for SAP BTP](#)

To check out different use cases and options, look at our samples: [Repository with Samples for the Terraform Provider for SAP BTP](#).

For blog posts in SAP Community, search for "infrastructure as code" in the [Technology Blogs by SAP](#).

## 6.2 Administration and Operations in the Cloud Foundry Environment

Learn about the different account administration and application operation tasks which you can perform in the Cloud Foundry environment.

- [Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)
- [Account Administration in the Cockpit \[page 2140\]](#)
- [Account Administration Using APIs of the SAP Cloud Management Service \[page 2378\]](#)
- [About Roles in the Cloud Foundry Environment \[page 2431\]](#)
- [Org Administration Using the Cockpit \[page 2433\]](#)
- [Org Management Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2423\]](#)
- [Org Administration Using the Cloud Foundry CLI \[page 2451\]](#)
- [Application Operations in the Cloud Foundry Environment \[page 2490\]](#)
- [Log on with a Browser to the Cloud Foundry CLI and Service Dashboards \[page 2256\]](#)
- [Accessing Administration Using APIs of the SAP Authorization and Trust Management Service \[page 2416\]](#)
- [Audit Logging in the Cloud Foundry Environment \[page 2501\]](#)

### Related Information

[Cloud Foundry Environment \[page 49\]](#)

[Getting Started in the Cloud Foundry Environment \[page 147\]](#)

[Development in the Cloud Foundry Environment \[page 233\]](#)

## 6.2.1 Org Management Using the SAP BTP Command Line Interface (btp CLI)

The Cloud Foundry environment allows you to create polyglot cloud applications in Cloud Foundry. To manage the lifecycle of an org in the Cloud Foundry environment, use the accounts/environment-instance command in the btp CLI.

### Available Plans

To create a Cloud Foundry org in a subaccount, you use the Cloud Foundry Runtime service. The Cloud Foundry Runtime service offers the following plans for this purpose:

Plan Name	Description
standard	<p>This is an enterprise-grade plan that allows you to create an org in your Cloud Foundry environment to start developing polyglot cloud-native applications.</p> <p>By default, a Cloud Foundry org that is created with this plan does not have any application runtime. To assign application runtime, the global account admin must assign the Cloud Foundry Runtime service with the MEMORY plan to the subaccount so that the org has memory for its applications.</p>
free	<p>Use this plan to try out the Cloud Foundry environment without an additional charge before switching to the standard plan that supports enterprise-grade productivity.</p> <p>By default, a Cloud Foundry org that is created with this plan is given a fixed application-runtime quota. There is no need to assign the Cloud Foundry Runtime service with the MEMORY plan to the subaccount as it will have no effect on the org quota.</p> <p>For more information about the free tier model for SAP BTP and its availability, see <a href="#">Using Free Service Plans [page 91]</a>.</p>
trial	<p>Use this plan to utilize your trial period.</p> <div style="background-color: #f0f0f0; padding: 10px; border-radius: 5px;"><p><b>Note</b></p><p>Your subaccount's type must be trial to use this plan.</p></div>

For more information about the features that each plan offers, see [Cloud Foundry Runtime](#) on SAP Discovery Center.

## Command Reference for CRUD Operations

The btp CLI offers the following environment instance command actions to manage Cloud Foundry orgs in SAP BTP:

Action	Command	Description	Parameters	Additional Info
list	btp list accounts/ environment- instance -- subaccount  <code>&lt;SUBACCOUNT_ID&gt;</code>	List all the Cloud Foundry orgs and other environment in- stances in a subac- count.	SUBACCOUNT_ID:  The ID of the subac- count.	
get	btp get accounts/ environment- instance <ID> --subaccount  <code>&lt;SUBACCOUNT_ID&gt;</code>	Get the details of a specific Cloud Foundry org in a subaccount.	ID: The ID of the en- vironment instance to view.  SUBACCOUNT_ID:  The ID of the subac- count.	

Action	Command	Description	Parameters	Additional Info
create	<pre>btp create accounts/ environment- instance -- subaccount &lt;SUBACCOUNT_ID&gt; --display-name &lt;DISPLAY_NAME&gt; --service &lt;SERVICE&gt; -- plan &lt;PLAN&gt; -- environment cloudfoundry -- parameters "{"instance_na me\":"&lt;ORG_NAM E&gt;"}"</pre>	<p>Create a Cloud Foundry org in a sub-account.</p> <p><b>SUBACCOUNT_ID:</b> The ID of the subaccount in which to create the Cloud Foundry org.</p> <p><b>DISPLAY_NAME:</b> The SAP BTP name of the environment instance.</p> <p><b>PLAN:</b> Specify either standard or free as the value.</p> <p><b>ORG_NAME:</b> The name of the org in the Cloud Foundry environment. Spaces are not allowed in the org name. Once the org is created, you cannot change its name.</p>		<p>For examples that show how to pass JSON parameters in the command line with different operating systems and shells, see the <a href="#">Passing JSON Parameters on the Command Line [page 2331]</a>.</p>

#### ⓘ Note

If the logged-in region has multiple landscapes, then you must specify the name of the landscape using the --landscape parameter in the command. You can obtain the valid values using the `btp list accounts/available-environment` command.

Action	Command	Description	Parameters	Additional Info
update	btp update accounts/environment-instance <ID> --subaccount <SUBACCOUNT_ID> --plan <PLAN>	Update an existing Cloud Foundry org in a subaccount.	ID: The ID of the environment instance to update  SUBACCOUNT_ID: The ID of the subaccount.  PLAN: Specify either standard or free as the value.	For the Cloud Foundry environment, you can use this command to change only the plan of an existing Cloud Foundry environment instance. In other words, update an org created with the free plan of the Cloud Foundry Runtime to the standard plan (not vice versa).
delete	btp delete accounts/environment-instance <ID> --subaccount <SUBACCOUNT_ID>	Delete a Cloud Foundry org created in a subaccount.	ID: The ID of the environment instance to delete.  SUBACCOUNT_ID: The ID of the subaccount where the environment instance exists.	The Cloud Foundry org and all its data will be lost.

#### ⓘ Note

Before updating the plan to standard, make sure that the MEMORY plan for the Cloud Foundry Runtime entitlement is assigned to your subaccount, and that you've assigned sufficient application runtime quota to cover the current workload in the subaccount.

### Note

You can obtain the subaccount ID by running `btp list accounts/subaccount`. You can also target the subaccount using `btp target --subaccount <ID>`, and then omit the `subaccount` parameter from the rest of the commands.

## Adding and Removing Org Managers

You can also use the btp CLI to add or remove org managers.

Action	Command	Description	Parameters	Additional Info
update	<p>For the default identity provider:</p> <pre>btp update accounts/ environment-instance &lt;ID&gt; --subaccount &lt;SUBACCOUNT_ID&gt; --plan &lt;PLAN&gt; --parameters "{"usersToAdd": [   {"id": "myUserID", "email": "name@example.com"} ]}</pre> <p>For a custom identity provider:</p> <pre>btp update accounts/ environment-instance &lt;ID&gt; --subaccount &lt;SUBACCOUNT_ID&gt; --plan &lt;PLAN&gt; --parameters "{"usersToAdd": [   {"id": "myUserID", "email": "name@example.com", "origin": "&lt;your custom IDP&gt;"} ]}</pre>	<p>Add one or more org managers to a Cloud Foundry org in a sub-account.</p> <p>usersToAdd adds the OrgManager role to the user, and, if it doesn't exist yet, it creates the user in the org.</p>	<p>ID: The ID of the environment instance for which to add org managers.</p> <p>SUBACCOUNT_ID: The ID of the subaccount.</p> <p>PLAN: Specify either standard or free as the value.</p>	<p>For examples that show how to pass JSON parameters in the command line with different operating systems and shells, see the <a href="#">Passing JSON Parameters on the Command Line [page 2331]</a>.</p>

Action	Command	Description	Parameters	Additional Info
update	<p>For the default identity provider:</p> <pre>btp update accounts/ environment-instance &lt;ID&gt; --subaccount &lt;SUBACCOUNT_ID&gt; --plan &lt;PLAN&gt; --parameters "{"usersToRemove": [   {"id": "myUserID", "email": "name@example.com"}]}</pre> <p>For a custom identity provider:</p> <pre>btp update accounts/ environment-instance &lt;ID&gt; --subaccount &lt;SUBACCOUNT_ID&gt; --plan &lt;PLAN&gt; --parameters "{"usersToRemove": [   {"id": "myUserID", "email": "name@example.com", "origin": "&lt;your custom IDP&gt;"}]}</pre>	<p>Remove one or more org managers from a Cloud Foundry org in a subaccount.</p> <p>usersToRemove removes the OrgManager role from the user, but it doesn't remove the user from the org.</p>	<p>ID: The ID of the environment instance for which to remove org managers.</p> <p>SUBACCOUNT_ID: The ID of the subaccount.</p> <p>PLAN: Specify either standard or free as the value.</p>	<p>To learn how to remove a user from the Cloud Foundry org and the subaccount, see <a href="#">About User Management in the Cloud Foundry Environment [page 2430]</a>.</p>

## Related Information

[Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

[Org Administration Using the Cockpit \[page 2433\]](#)

[Add Org Members \[page 2437\]](#)

[Managing Spaces \[page 2440\]](#)

[About User Management in the Cloud Foundry Environment \[page 2430\]](#)

## 6.2.2 About User Management in the Cloud Foundry Environment

The Cloud Foundry environment has its own store for user data within SAP BTP. Understanding the relationship between SAP BTP and the Cloud Foundry environment is useful.

All users of Cloud Foundry and SAP BTP are stored in identity providers. Both require a local copy of the users so that you can assign authorizations to those users. When a user tries to log on, the user is forwarded to the identity provider to authenticate. When you add a user as a member to an org or space, there's no check whether or not this user actually exists in the identity provider you specified.

Org members can only be added by an `Org Manager`. If you only have the `Space Manager` role, you can only add space members that are known to the org. To do that, you must first ask your `Org Manager` to add the users as org members with no roles.

### ⓘ Note

All org members implicitly have the `Org User` role.

If you're the `Org Manager`, you already have the authorizations to add users to the org. When you add a new user as a space member, that user is added automatically as part of the org as well.

## Cloud Foundry and Shadow Users

Whenever you add a user at the org or space level with the SAP BTP cockpit, SAP BTP also creates the user at the subaccount level, also known as a shadow user. When you use the SAP BTP cockpit to add a user, SAP BTP assigns at least one Cloud Foundry role to the user, namely `Org User` if you don't add any other role. SAP BTP checks for Cloud Foundry users with a role and creates corresponding shadow users at the subaccount level if no such user exists. Creating users with the Cloud Foundry CLI or Cloud Foundry APIs doesn't add any Cloud Foundry roles by default. Once you add a role to the user, the system ensures that there's a corresponding user at the subaccount level.

### → Remember

When you delete a user at the Cloud Foundry level, SAP BTP doesn't delete the corresponding shadow user at the subaccount level.

When you delete a user at the subaccount level, delete the user at the Cloud Foundry org and space levels, too. Otherwise, the system automatically creates the corresponding user at the subaccount level if the user has a Cloud Foundry role.

For more information about shadow users, see [Working with Users \[page 2265\]](#).

## User Provisioning

For user management of Cloud Foundry organizations, use the provisioning connectors of SAP Cloud Identity Services - Identity Provisioning. For more information, see [SAP BTP Integration Scenario](#).

## Related Information

[User and Member Management \[page 104\]](#)

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

[Add Org Members \[page 2437\]](#)

[Add Organization Members Using the Cloud Foundry Command Line Interface \[page 2483\]](#)

[Add Space Members \[page 2444\]](#)

[Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#)

### 6.2.3 About Roles in the Cloud Foundry Environment

Roles determine which features users can view and access, and which actions they can initiate.

Cloud Foundry includes predefined roles that are specific to the navigation level in the SAP BTP cockpit; for example, the roles at the level of the organization differ from the ones for the space. Each role comes with a set of permissions. Roles apply to all operations that are associated with the organization or the space, irrespective of the tool used (Eclipse-based tools, SAP BTP cockpit, and cf CLI).

The following roles can be assigned to users in the Cloud Foundry environment on SAP BTP:

Level	Role	Role Description
Organization	Org Manager	Manages the organization. When you create an org, you get the Org Manager role in that org by default.
	Org Auditor	Provides read-only access to user information and org quota usage information.

Level	Role	Role Description
	Org User	<p>Provides read-only access to the list of other organization users and their roles.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"> <b> ⓘ Note</b> <p>When creating an org member, you can choose one or more of the following roles:</p> <ul style="list-style-type: none"> <li>• Org Manager</li> <li>• Org User</li> <li>• Org Auditor</li> </ul> <p>When creating a space member, they get the Org User role by default.</p> </div>
Space	Space Manager	Manages a space within an organization.
	Space Developer	<p>Manages applications, services, and space-scoped service brokers in a space.</p> <div style="border-left: 3px solid #C0392B; padding-left: 10px;"> <b> ⚠ Caution</b> <p>The Space Developer role has broad rights within Cloud Foundry and, in particular, has access to the credentials used in various services and app bindings as well as other sensitive data. For more information, see <a href="#">Giving Access Rights to Platform Users</a>.</p> </div>
	Space Auditor	Provides read-only access to a space.
	Space Supporter	<p>Allows you to troubleshoot and debug applications and service bindings in a space.</p> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"> <b> ⓘ Note</b> <p>To add the Space Supporter role using the <code>set-space-role</code> command, you need to have installed version 8 of the Cloud Foundry Command Line Interface (cf CLI). See <a href="https://docs.cloudfoundry.org/cf-cli/install-go-cli.html">https://docs.cloudfoundry.org/cf-cli/install-go-cli.html</a>.</p> </div>

### ⓘ Note

Managing members and roles for an organization takes place on the navigation level of the subaccount to which the organization is assigned.

For more information about organization and space roles in Cloud Foundry, see <https://docs.cloudfoundry.org/concepts/roles.html>.

## Related Information

[About User Management in the Cloud Foundry Environment \[page 2430\]](#)

[Add Org Members \[page 2437\]](#)

[Add Organization Members Using the Cloud Foundry Command Line Interface \[page 2483\]](#)

[Add Space Members \[page 2444\]](#)

[Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#)

## 6.2.4 Org Administration Using the Cockpit

In the Cloud Foundry environment, manage orgs, spaces and space quota plans using the SAP BTP cockpit.

In addition to global accounts and subaccounts, the Cloud Foundry environment includes another hierarchical level represented by orgs and spaces.

When you enable Cloud Foundry in a subaccount, you create a Cloud Foundry org in which you can then create multiple spaces. Each subaccount can contain only one Cloud Foundry org. There is no limit to how many spaces you can have within one org.

In the Cloud Foundry environment, you deploy applications and consume services at space level. Similar to subaccounts, spaces enable you to once again structure and sub-divide quota if you want to. You can do this by managing space quota plans.

- [Managing Orgs \[page 2434\]](#)
- [Managing Spaces \[page 2440\]](#)
- [Managing Space Quotas \[page 2447\]](#)

### 6.2.4.1 Navigate to Orgs and Spaces

To administer your Cloud Foundry environment, navigate to orgs, and spaces in the SAP BTP cockpit.

## Prerequisites

- Sign up for an enterprise or a trial account and receive your logon data.  
For more information, see [Get a Free Trial \[page 144\]](#) or [Sign up for a Customer Account \[page 145\]](#).
- Create the org or space to which you want to navigate.  
For more information, see [Create Orgs \[page 2435\]](#) and [Create Spaces \[page 2441\]](#).

## Procedure

Entity You'd Like to Navigate to	Navigation Path
Org	<p>Navigate to the subaccount that contains the Cloud Foundry org. If you've already enabled the Cloud Foundry environment in your subaccount, you see the name of your organization, the number of its spaces and members, and its API endpoint on the <a href="#">Overview</a> page of your subaccount. If you haven't enabled Cloud Foundry yet, choose <a href="#">Enable Cloud Foundry</a> to create a Cloud Foundry org.</p> <div style="background-color: #f0f0f0; padding: 10px; border-left: 3px solid #ccc;"><p><b> ⓘ Note</b></p><p>Note that your subaccount and your org have a 1:1 relationship. They have the same name and therefore also the same navigation level in the cockpit.</p></div>
Space	<p>Navigate to the subaccount that contains the space you'd like to navigate to.</p> <ol style="list-style-type: none"><li>On the subaccount <a href="#">Overview</a> page, you have table with all your spaces. Choose the space you want to navigate to.</li><li>In the navigation area, choose  <a href="#">Cloud Foundry</a>  <a href="#">Spaces</a> </li></ol>

## Related Information

[Navigate in the Cockpit \[page 2142\]](#)

### 6.2.4.2 Managing Orgs

In the SAP BTP cockpit, learn how to create a Cloud Foundry org, manage its members, and later delete it, if needed.

- [Create Orgs \[page 2435\]](#)
- [Delete Orgs \[page 2436\]](#)
- [Managing Org Members \[page 2436\]](#)

**ⓘ Note**

If you work with the Cloud Foundry CLI (cf CLI), be aware that you are responsible for all actions performed using the CLI. Also, you won't always be able to mirror the same data in the SAP BTP cockpit.

## 6.2.4.2.1 Create Orgs

To enable Cloud Foundry in your subaccount, you must create an org in order to use it.

### Procedure

1. Navigate to your subaccount *Overview* page in the SAP BTP cockpit.
2. Choose *Enable Cloud Foundry*.

After you choose *Enable Cloud Foundry*, follow the steps provided by the wizard to complete the creation of your org.

3. Choose a plan from the drop-down list and enter a name for your org.

#### Note

Once you've created the org, you can't change its name afterwards in the SAP BTP cockpit.

As an optional step, you can customize the instance name of your environment. Otherwise, the wizard uses the automatically generated instance name.

If you decide to enter an environment instance name of your choice, make sure that it's CLI-friendly. A CLI-friendly name is a string of up to 32 characters: alphanumeric (A-Z, a-z, 0-9), periods, underscores, and hyphens. Having a CLI-friendly name allows you to use it in btp CLI commands.

4. If the region you're working in has multiple landscapes, the *Landscape* option is displayed. Choose the landscape in which to create the org.
5. Choose *Create*.

### Related Information

[Navigate to Orgs and Spaces \[page 2433\]](#)

[Create a Subaccount \[page 2173\]](#)

## 6.2.4.2.2 Delete Orgs

You can delete a Cloud Foundry org from a subaccount using the SAP BTP cockpit. Once the org is deleted, you can create a new one.

### Prerequisites

You are a global account administrator, as well as a member of the subaccount containing the org you want to delete.

#### ⓘ Note

You can delete an org using only the cockpit or the btp CLI. You cannot use the Cloud Foundry CLI to perform this task.

### Procedure

1. Navigate to the subaccount containing the org.
2. Choose *Disable Cloud Foundry* and confirm the operation.

### Results

The org is deleted. All data in the org including spaces, applications, service instances, and member information is lost. You can now choose *Enable Cloud Foundry* to create a new org in the subaccount.

### Related Information

[Navigate to Orgs and Spaces \[page 2433\]](#)

## 6.2.4.3 Managing Org Members

Learn how to add, edit, and delete org members in the SAP BTP cockpit.

- [Add Org Members \[page 2437\]](#)
- [Edit Org Members \[page 2438\]](#)
- [Delete Org Members \[page 2439\]](#)

### 6.2.4.3.1 Add Org Members

In the SAP BTP cockpit, add users as org members and assign roles to grant the users access to information, such as user and quota information in a Cloud Foundry org.

#### Prerequisites

- You have the Org Manager role for the org in question.

 Note

When you enable Cloud Foundry and the org is created, the Org Manager role is assigned automatically to you.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

- The users exist in a trusted platform identity provider.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

#### Procedure

1. Navigate to your subaccount in the cockpit. Make sure that Cloud Foundry is enabled. For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).
2. In the navigation area, choose  . The screen displays all members currently assigned to the org in a list.
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses.  
Use commas (,), spaces ( ), semicolons (;), or line breaks to separate members.
5. Enter the [Origin](#) for the identity provider, which hosts the members you just added.  
The default identity provider is `sap.ids`. For more information, see [Default Identity Provider \[page 2258\]](#).  
If the new members are platform users from a custom identity provider, enter the origin.  
Select the corresponding origin of the Identity Authentication tenant.

 Note

The option to select an origin appears only if you have two or more identity providers.

For more information about finding the origin, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).

6. Select the org roles you want to assign.  
For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).
7. Choose [Add](#).

## Next Steps

- To change the roles of a member, see [Edit Org Members \[page 2438\]](#).
- To remove all the roles of a member, see [Delete Org Members \[page 2439\]](#).  
This action removes the member from the organization and any spaces in the org.
- If the users need to view or manage applications in spaces, add the users to the relevant spaces, too.  
For more information, see [Add Space Members \[page 2444\]](#).

## Related Information

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Add Organization Members Using the Cloud Foundry Command Line Interface \[page 2483\]](#)

### 6.2.4.3.2 Edit Org Members

Remove existing roles or assign new roles to an org member in the SAP BTP cockpit.

#### Prerequisites

- You have the Org Manager role for the org in question.

##### ⓘ Note

When you enable Cloud Foundry and the org is created, the Org Manager role is assigned automatically to you.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

- The users exist in a trusted platform identity provider.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

#### Procedure

1. In the navigation area in the cockpit, choose . The screen displays all members currently assigned to the org in a list.
2. Find the org member in the list and choose .
3. Select or unselect the roles you want to assign or remove.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

4. Choose [Save](#).

The changes you make to the roles of a member take effect immediately.

## Related Information

[Add Org Members Using the Cockpit \[page 2437\]](#)

[Delete Org Members Using the Cockpit \[page 2439\]](#)

### 6.2.4.3.3 Delete Org Members

Delete all roles of an org member in the SAP BTP cockpit. As a consequence, the member is removed from the org and any spaces in that org.

#### Prerequisites

- You have the Org Manager role for the org in question.

##### Note

When you enable Cloud Foundry and the org is created, the Org Manager role is assigned automatically to you.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

- The users exist in a trusted platform identity provider.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

#### Procedure

1. In the navigation area in the cockpit, choose   [Cloud Foundry](#)  [Org Members](#).  
The screen displays all members currently assigned to the org in a list.
2. Find the org member in the list. To remove all the roles of that member, choose  ([Delete](#)).

This action removes the member from the organization and any spaces in the org.

## Related Information

[Managing Org Members \[page 2436\]](#)

## 6.2.4.4 Managing Spaces

Learn what a Cloud Foundry space is, what type of information you will find on the [Spaces](#) page in the SAP BTP cockpit, and what you can do with or within a space.

### What Is a Space?

A space is a shared location on the platform in which you can deploy and maintain applications, and connect them to services. You can create multiple spaces within one org. There is no limit to how many spaces you can have in a single org.

By default, a space inherits the org quota. If you don't want your space to consume the entire org quota, you can use space quota instead. For more information, see [Managing Space Quotas \[page 2447\]](#).

### Prerequisites

If you want to see any spaces on the [Spaces](#) page in the cockpit, you must have either the Org Manager role or any one of the space roles.

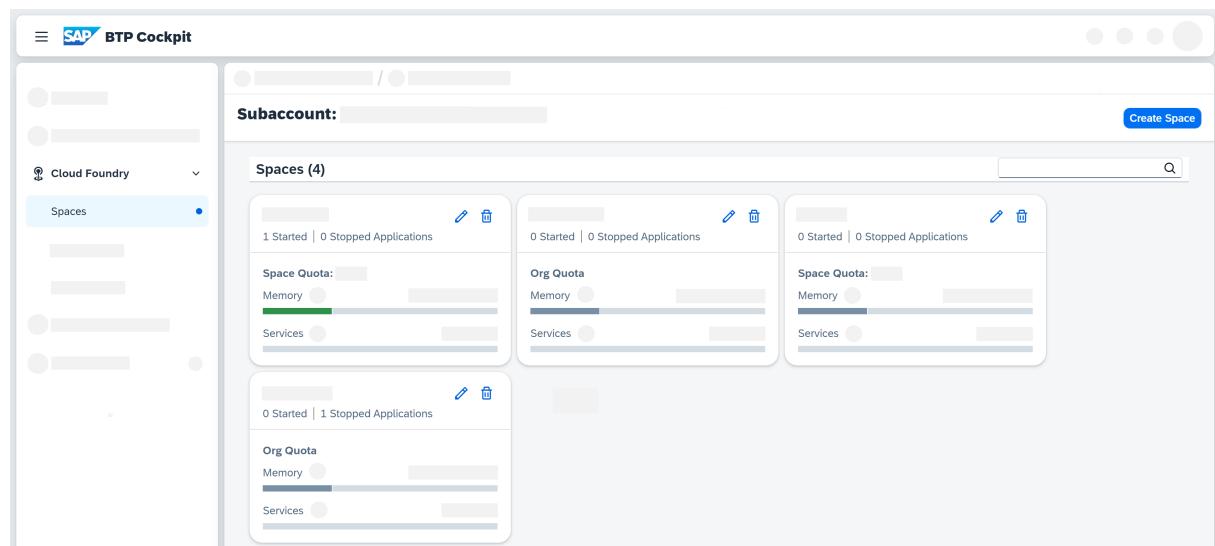
The Org Manager role gives you access to all spaces in an org. A space role gives you access to a specific space.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

### What Can You Find on the Spaces Page?

To visit this page in the cockpit:

1. Navigate to a subaccount of your choice.
2. Navigate to the page.



You find the following on this page:

- A list of all spaces you have access to.
  - Details about each space tile, such as:
    - Number of started and stopped applications in a space
    - Name of the space quota if a space quota is assigned to the space, or *Org Quota* if the space is currently using the org quota
- To manage these settings, navigate to  [Cloud Foundry](#)  [Space Quotas](#)  [Quota Assignments](#). For more information, see [Assign Space Quotas to Spaces \[page 2450\]](#) and [Edit Space Quotas \[page 2451\]](#).
- Available resources such as memory and services with their upper consumption limit and current usage statuses

## What Can You Do With or Within a Space?

Here are some of the space-related actions you can do:

- [Create Spaces \[page 2441\]](#)
- [Managing Space Members \[page 2443\]](#)
- [Deploy an Application \[page 2491\]](#)
- Edit Spaces  
To edit a space, choose  *(Edit) on the space tile. This option lets you change the name of the space.*  
Org Managers and Space Managers are allowed to rename spaces.
- [Delete Spaces \[page 2443\]](#)

## Related Information

[Org Administration Using the Cockpit \[page 2433\]](#)

[Org Administration Using the Cloud Foundry CLI \[page 2451\]](#)

### 6.2.4.4.1 Create Spaces

Create spaces in your Cloud Foundry organization using the SAP BTP cockpit. In a space, you can deploy and maintain applications, and connect them to services.

## Prerequisites

You have a Cloud Foundry organization in your subaccount, and you have the Org Manager role in that organization.

See [Create Orgs \[page 2435\]](#) and [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

## Context

There are two ways to create a space in the cockpit. The main one is to do it from the *Spaces* page in your subaccount. The second one is from the *Overview* page of your subaccount. You can choose either one of them.

## Procedure

Create a Space from the Spaces Page	Create a Space from the Subaccount Overview Page
<ol style="list-style-type: none"><li>1. Navigate to the subaccount that contains the Cloud Foundry organization in which you'd like to create a space.</li><li>2. Navigate to the  <i>Cloud Foundry</i>  <i>Spaces</i>  page.</li><li>3. Choose <i>Create Space</i> from the top right hand-side corner of that page.</li><li>4. Enter a space name and choose the space roles you want to assign to yourself.</li><li>5. Then, choose <i>Save</i>.</li></ol>	<ol style="list-style-type: none"><li>1. Navigate to the subaccount that contains the Cloud Foundry organization in which you'd like to create a space.</li><li>2. On the subaccount <i>Overview</i> page, you have a <i>Spaces</i> table. Choose <i>Create Space</i> from the top right hand-side corner of that table.</li><li>3. Enter a space name and choose the space roles you want to assign to yourself.</li><li>4. Then, choose <i>Save</i>.</li></ol>

## Next Steps

As an optional step, you can assign space quotas to spaces. If you don't assign a space quota, the space you've just created will use the consumption limits of the org quota.

For more information, see [Assign Space Quotas to Spaces \[page 2450\]](#).

## Related Information

[Create Spaces Using the Cloud Foundry Command Line Interface \[page 2485\]](#)

## 6.2.4.4.2 Delete Spaces

Delete spaces in your Cloud Foundry org using the SAP BTP cockpit.

### Prerequisites

- You have the Org Manager role in the org from which you want to delete a space. For more information about roles and permissions, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).
- You've ensured that the data in the space you're going to delete is no longer needed.

#### ⚠ Caution

Please be aware that you won't be able to access your SAP HANA database system if you delete a space in an organization to which an SAP HANA database system is assigned.

### Procedure

1. Navigate to  [Cloud Foundry](#)  in your Cloud Foundry subaccount and choose  ([Delete](#)) on the tile of the space you want to delete.
2. Confirm your change.

### Related Information

[Create Spaces Using the Cockpit \[page 2441\]](#)

[Delete Spaces Using the Cloud Foundry Command Line Interface \[page 2490\]](#)

## 6.2.4.5 Managing Space Members

Learn how to add, edit, and delete space members in the SAP BTP cockpit.

- [Add Space Members \[page 2444\]](#)
- [Edit Space Members \[page 2445\]](#)
- [Delete Space Members \[page 2446\]](#)

## 6.2.4.5.1 Add Space Members

You can add space members and assign roles to them at the space level in the SAP BTP cockpit.

### Prerequisites

- You have the Space Manager or Org Manager role.

If you only have the Space Manager role, the users you want to add to the space must already have the Org User role.

If you have the Org Manager role, you can assign space roles to users even if they're not members of the org yet.

#### ⓘ Note

When you add users who aren't yet members of an org to a space in that org, they get the Org User role by default. This makes them members of the org as well.

- The users exist in a trusted platform identity provider.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

### Procedure

1. Navigate to   in the SAP BTP cockpit. For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).

2. Choose a space to which you'd like to add members. For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).

3. In the navigation area, choose *Space Members*.

The screen displays all members currently assigned to the space in a list.

4. Choose *Add Members*.

5. Enter one or more e-mail addresses.

Use commas (,), spaces ( ), semicolons (;), or line breaks to separate members. The changes you make to the roles of a member take effect immediately.

6. Enter the *Origin* for the identity provider, which hosts the members you just added.

The default identity provider is `sap.ids`.

For more information, see [Default Identity Provider \[page 2258\]](#).

If the new members are platform users from a custom identity provider, enter the origin.

Select the corresponding origin of the Identity Authentication tenant.

For more information about finding the origin, see [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#).

7. Select the space roles you want to assign.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

8. Choose *Add*.

## Next Steps

You also have the following options:

- [Edit Space Members \[page 2445\]](#)
- [Delete Space Members \[page 2446\]](#)

## Related Information

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

[Add Org Members Using the Cockpit \[page 2437\]](#)

[Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#)

### 6.2.4.5.2 Edit Space Members

Remove existing roles or assign new roles to a space member in the SAP BTP cockpit.

#### Prerequisites

- You have the Space Manager or Org Manager role.
- The users exist in a trusted platform identity provider.

For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

#### Procedure

1. Navigate to   in the SAP BTP cockpit. For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).
2. Choose the space in which you'd like to edit the space roles of any of its members.
3. In the navigation area, choose *Space Members*.  
The screen displays all members currently assigned to the space in a list.
4. Choose .
5. Select or unselect the space roles you want to assign or remove.

For more information, see [About Roles in the Cloud Foundry Environment \[page 2431\]](#).

6. Choose [Save](#).

The changes you make to the roles of a member take effect immediately.

## Related Information

[Add Space Members \[page 2444\]](#)

[Delete Space Members \[page 2446\]](#)

### 6.2.4.5.3 Delete Space Members

Delete all roles of a space member in the SAP BTP cockpit. As a consequence, the member is removed from the space.

#### Prerequisites

- You have the Space Manager or Org Manager role.
- The users exist in a trusted platform identity provider.  
For more information, see [About User Management in the Cloud Foundry Environment \[page 2430\]](#).

#### Procedure

1. Navigate to   in the SAP BTP cockpit. For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).
2. Choose the space from which you'd like to remove members.
3. In the navigation area, choose [Space Members](#).  
The screen displays all members currently assigned to the space in a list.
4. To remove all the roles of a member, choose  ([Delete](#)).  
This action removes the member from the space.

## Related Information

[Managing Space Members \[page 2443\]](#)

## 6.2.4.6 Managing Space Quotas

Learn what a space quota is, what type of information you will find on the [Space Quotas](#) page in the SAP BTP cockpit, and what you can do with a space quota in the Cloud Foundry environment.

### What Is a Space Quota?

A space quota defines the upper consumption limit for all resources in a space, such as:

- Memory
- Route
- Services
- Instance memory
- Application instances

#### ⓘ Note

Assigning a space quota to a space is optional. A space without an assigned space quota shares the consumption limits defined in the org quota with all the other spaces part of the org.

When working with space quotas, keep in mind the following additional specifics:

- You can create unlimited space quotas in an organization (org), but you can assign only one space quota to a space at a time.
- If a space quota is larger than the org quota, the org quota limit is applied.  
Example: If you create a space quota with memory of 2048MB but the org quota has a total memory of 1024MB, then the space is going to use the org quota instead.
- Also, you can have only one space quota named "dev" in an org, but two orgs can have two separate space quotas with the same name "dev".
- When creating a space quota, you can define whether a space quota allows you to use paid services within a space.

### Prerequisites

To create, edit, or assign space quotas in an org, you must have the Org Manager role.

### What Can You Find on the Space Quotas Page?

You can switch between the following views on this page:

- [Space Quotas](#)

Space Quota	Memory	Paid Services	Routes	App Instances	Services	Instance Memory
1	0MB	Not Allowed	0	Unlimited	20	Unlimited
2	0MB	Allowed	0	0	500	0MB

You find the following in this view:

- A list of all space quotas you have created and whether they are assigned to a space.
- Details about each space quota tile related to the space, such as:
  - *Memory (MB)*: Total amount of memory
  - *Routes*: Total number of routes
  - *Services*: Total number of service instances
  - *Paid Services*: Selected if you'd like to allow the provisioning of instances of paid service plans
  - *App Instances*: Total number of application instances
  - *Instance Memory (MB)*: Maximum amount of memory an application instance can have (-1 represents an unlimited amount)
- The option to create a space quota.  
For more information, see [Create Space Quotas \[page 2449\]](#).
- The option to edit a space quota.  
For more information, see [Edit Space Quotas \[page 2451\]](#).
- The option to delete a space quota.  
You can delete a space quota only if it's not assigned to a space.

- *Quota Assignments*

Space	Space Quota	Actions
Space 1	Quota 1	<i>Edit</i>
Space 2	Quota 2	<i>Edit</i>
Space 3	Quota 3	<i>Edit</i>
Space 4	Quota 4	<i>Edit</i>
Space 5	Quota 5	<i>Edit</i>
Space 6	Quota 6	<i>Edit</i>

You find the following in this view:

- A list of all the available spaces in the subaccount you have access to and whether they have a space quota assigned to them or if they are using the org quota instead.
- The option to *Edit* a quota assignment.

For more information, see [Assign Space Quotas to Spaces \[page 2450\]](#).

## What Can You Do with a Space Quota?

- [Create Space Quotas \[page 2449\]](#)
- [Assign Space Quotas to Spaces \[page 2450\]](#)
- [Edit Space Quotas \[page 2451\]](#)

## Related Information

[Managing Spaces \[page 2440\]](#)

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

### 6.2.4.6.1 Create Space Quotas

You can use the SAP BTP cockpit to create space quotas.

#### Prerequisites

You have the Org Manager role for the org in which you want to create a space quota.

#### Procedure

1. Navigate to the subaccount that contains the spaces to which you want to add quotas.
2. In the navigation menu, choose [Cloud Foundry](#) [Space Quotas](#).
3. Choose [Create Space Quota](#).
4. Enter a name for your space quota and add limits for the following:
  - [\*Memory \(MB\)\*](#): Total amount of memory a space can have
  - [\*Routes\*](#): Total number of routes
  - [\*Services\*](#): Total number of service instances
  - [\*Instance Memory \(MB\)\*](#): Maximum amount of memory an application instance can have (-1 represents an unlimited amount)
  - [\*App Instances\*](#): Total number of application instances
  - (Optional) [\*Allow Paid Service\*](#): Switch on if you'd like to allow the provisioning of instances of paid service plans

#### Note

If the space quota limit for a resource exceeds the org quota limit, the org quota limit applies.

5. Choose [Create](#).

## Related Information

[Entitlements and Quotas \[page 100\]](#)

[Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#)

### 6.2.4.6.2 Assign Space Quotas to Spaces

You can use the SAP BTP cockpit to assign space quotas to spaces.

#### Prerequisites

- You have the Org Manager role for the org in which you want to create and assign a space quota.
- You have at least one space quota. For more information, see [Create Space Quotas \[page 2449\]](#).

#### Procedure

1. Navigate to the subaccount that contains the spaces to which you want to assign space quotas.
2. In the navigation menu, choose  [Cloud Foundry](#)  [Space Quotas](#).
3. Open the [Quota Assignments](#) tab.
4. Select one of the existing quota assignments and choose  [\(Edit\)](#) to make changes.
5. Select either a specific [Space Quota](#) from the drop-down or the [Org Quota](#).  
If you select [Org Quota](#), the space will use the consumption limits defined in the org quota.
6. To assign the selected space or org quota to the space, choose [Save](#).

## Related Information

[Entitlements and Quotas \[page 100\]](#)

[Create Space Quotas Using the Cockpit \[page 2449\]](#)

[Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 2488\]](#)

## 6.2.4.6.3 Edit Space Quotas

Edit space quotas in the Cloud Foundry environment using the SAP BTP cockpit.

### Prerequisites

You have the Org Manager role for the org in which you want to edit space quotas.

### Procedure

1. Choose the subaccount that contains the spaces for which you'd like to edit the space quota.
2. In the navigation menu, choose  [Cloud Foundry](#)  [Space Quotas](#).
3. Choose  [\(Edit\)](#) for an existing space quota.
4. Adjust the values in the text fields as needed.

For more information about the different values, see <https://docs.cloudfoundry.org/adminguide/quota-plans.html#space>.

5. Choose [Save](#).

### Related Information

[Change Space Quota Plans Using the Command Line Interface \[page 2489\]](#)

[Create Space Quotas \[page 2449\]](#)

[Assign Space Quotas to Spaces \[page 2450\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

[Entitlements and Quotas \[page 100\]](#)

## 6.2.5 Org Administration Using the Cloud Foundry CLI

Use the Cloud Foundry command line interface (CF CLI) for managing subaccounts in the Cloud Foundry environment, such as creating orgs and spaces, or managing quota.

- [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)
- [CF CLI: Plug-ins \[page 2454\]](#)
- [Create Spaces Using the Cloud Foundry Command Line Interface \[page 2485\]](#)
- [Add Organization Members Using the Cloud Foundry Command Line Interface \[page 2483\]](#)
- [Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#)

- [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#)
- [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 2488\]](#)
- [Change Space Quota Plans Using the Command Line Interface \[page 2489\]](#)

 **Note**

If you work with the Cloud Foundry CLI (cf CLI), be aware that you are responsible for all actions performed using the CLI. Also, you won't always be able to mirror the same data in the SAP BTP cockpit.

You must delete an org using only the cockpit or the SAP BTP command line interface (btp CLI).

## 6.2.5.1 Working with the Cloud Foundry Command Line Interface

Find out how to get and use the Cloud Foundry command line interface.

### Related Information

[Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

[Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#)

[CF CLI: Plug-ins \[page 2454\]](#)

### 6.2.5.1.1 Download and Install the Cloud Foundry Command Line Interface

Download and set up the Cloud Foundry Command Line Interface (cf CLI) to start working with the Cloud Foundry environment.

### Procedure

1. Download the latest version of cf CLI from GitHub at the following URL: <https://github.com/cloudfoundry/cli#downloads>
  - For Microsoft Windows, download the Windows 64-bit installer.
  - For Mac OS, download the PKG file. You can use the [Homebrew](#) open source package management software to download the latest available version of the cf CLI.
2. Install the cf CLI:
  - For Microsoft Windows, unpack the ZIP file.

- For Mac OS, proceed as follows:
  1. Open the PGK file.
  2. In the installation wizard, choose *Continue*, and then select the destination folder for the cf CLI installation.
  3. Choose *Continue*, and when prompted, choose *Install*.

For more information, see <http://docs.cloudfoundry.org/devguide/installcf/install-go-cli.html>.

3. (Optional) If you have an HTTP proxy server, configure the proxy settings. For more information, see <http://docs.cloudfoundry.org/cf-cli/http-proxy.html>.

## 6.2.5.1.2 Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface

Use the Cloud Foundry Command Line Interface (cf CLI) to log on to the Cloud Foundry space.

### Prerequisites

- The Cloud Foundry environment is enabled. You can enable it in the Cloud Platform Cockpit. For more information, see [Create a Subaccount \[page 2173\]](#).
- (Enterprise accounts only) You have created at least one subaccount and enabled the Cloud Foundry environment in this subaccount. For more information, see [Create a Subaccount \[page 2173\]](#).

#### ⓘ Note

In a Cloud Foundry trial account, the Cloud Foundry environment has been activated for you automatically and a first space "dev" has been created for you.

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).

### Procedure

1. Open a command line.
2. Set the target API endpoint to the cloud controller.

#### ⓘ Note

There is no specific endpoint for trial accounts. Both enterprise and trial accounts use the same API endpoints.

```
cf api https://api.cf.eu10.hana.ondemand.com
```

For more information, see [Regions and API Endpoints Available for the Cloud Foundry Environment \[page 20\]](#).

3. Log on to the Cloud Foundry environment:

```
cf login
```

 Note

- To log on with SAP Universal ID, use the option `--sso`. You need a browser in the logon process.  
For more information, see [Log On Manually With a Custom Identity Provider \[page 2252\]](#).  
Otherwise log on with the password associated with your account (S-user or P-user) in the default identity provider, SAP ID service.  
If you've forgotten this password and this user is associated with your SAP Universal ID user, reset your password.  
For more information, see SAP Note [3085908](#).
- To log on with a specific identity provider, use the option `--origin`.  
For more information, see [Log On as a Technical User With a Custom Identity Provider \[page 2254\]](#).

4. When prompted, enter your user credentials (email and password).
5. To view the help for the CLI, execute `cf help`, which lists the most common CLI commands with a short description, or `cf help -a`, which lists all commands. To get help for a specific command, execute `cf help <command>`.

### 6.2.5.1.3 CF CLI: Plug-ins

A list of additional commands that have been implemented as plug-ins to extend the base CF CLI client.

[Multitarget Application Commands for the Cloud Foundry Environment \[page 2456\]](#)

#### 6.2.5.1.3.1 Multitarget Application Plug-In for the Cloud Foundry Command Line Interface

Use the Multitarget application plug-in for the Cloud Foundry command line interface to deploy, remove, and view MTAs, among other possible operations.

Before using the extended commands in the Cloud Foundry environment, you need to install the MTA plug-in in the Cloud Foundry environment as described in [Install the MultiApps CLI Plugin in the Cloud Foundry Environment \[page 2455\]](#).

### Related Information

[Multitarget Applications in the Cloud Foundry Environment \[page 570\]](#)

## 6.2.5.1.3.1.1 Install the MultiApps CLI Plugin in the Cloud Foundry Environment

The MultiApps command line interface plugin (formerly known as the MTA plugin) for the Cloud Foundry command line interface lets you deploy, remove, and view MTAs, among other possible operations, by extending Cloud Foundry commands.

### Prerequisites

You have installed the Cloud Foundry command line interface version 6.40 or higher. See [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).

### Procedure

1. Open the command line interface or terminal.
2. Check if a previous version is installed by using the command `cf plugins`. If the `MtaPlugin` is already installed, you have to uninstall it using the command `cf uninstall-plugin MtaPlugin`.

If you do not uninstall the previous version of the plugin and try to install a new one, you receive the following error:

#### ↔ Output Code

```
Plugin multiapps v<version> could not be installed as it contains commands
with names and aliases that are already used:
bg-deploy, deploy, download-mta-op-logs, mta, mta-ops, mtas, purge-mta-
config, undeploy, dmol
```

3. To install the latest available version of the plugin, proceed as follows:
  1. Make sure that you have the Cloud Foundry community repository in your Cloud Foundry command line interface. If it is not available there, add it by executing the following command:  
`cf add-plugin-repo CF-Community https://plugins.cloudfoundry.org`
  2. To install the plugin, enter the following command:  
`cf install-plugin multiapps -f`

#### ⓘ Note

Alternatively, if you want to install a specific version of the plugin, proceed as follows:

1. Download the [preferred version](#) of the plugin that is compatible with your operating system.
2. Extract the downloaded archive if required.
3. To install the plugin, enter the following command:
  - Mac and Linux: `cf install-plugin<path to the downloaded plugin> -f`
  - Windows: `cf install-plugin<path to the downloaded plugin> -f`

- Verify that the plugin has been installed successfully by entering `cf plugins`.

You see a list of plugins that now includes the MultiApps CLI Plugin for the Cloud Foundry command line interface. The output also displays commands that are specific to this plugin.

## Related Information

[SAP BTP, Cloud Foundry CLI Plugins](#)

[Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

[MultiApps CLI Plugin](#)

### 6.2.5.1.3.1.2 Multitarget Application Commands for the Cloud Foundry Environment

A list of additional commands to deploy multitarget applications (MTA) to the Cloud Foundry environment.

#### ⓘ Note

The expiration time for all MTA operations in Cloud Foundry is 3 days. If an operation is still active when time limit is reached, it is automatically aborted.

#### ⚠ Caution

Due to the missing shared domains in certain regions, as China (Shanghai), the URL of the SAP Cloud Deployment service should be specified by each multitarget application developer who uses MultiApps CF CLI plugin older than 3.0.0. If you are using MultiApps CF CLI plugin newer than 3.0.0, you do not need additional settings. You can do this by using the environment variable `MULTIAPPS_CONTROLLER_URL`, or the `-u` option specified in the commands listed below.

The URL of the deploy-service that needs to be set is in the following format: `deploy-service.cf.<domain>` The domain is derived from the Cloud Foundry API endpoint that you can find in the SAP BTP Cockpit in the Overview of your subaccount. For more information, see [Deploying Applications in Regions \[page 17\]](#) and [Regions and API Endpoints Available for the Cloud Foundry Environment \[page 20\]](#).

If you are using the `-u` option, make sure you have the MultiApps CLI Plugin version 2.1.3 or higher

If you are using region CN40 with API Endpoint `api.cf.cn40.platform.sapcloud.cn`, you need to specify the URL as follows: `export MULTIAPPS_CONTROLLER_URL=deploy-service.cf.cn40.platform.sapcloud.cn`

Commands for the Cloud Foundry Environment Overview

Command	Alias	Description
<a href="#">deploy</a>		Deploy a new Multitarget Application (MTA) or synchronize changes to an existing MTA

Command	Alias	Description
<code>bg-deploy</code>		Deploy a Multitarget Application using “blue-green” (zero-downtime) deployment
<code>undeploy</code>		Undeploy a Multitarget Application (MTA)
<code>mta</code>		Display information about a deployed Multitarget Application (MTA)
<code>mtas</code>		List all deployed Multitarget Applications (MTA)
<code>mta-ops</code>		List all active operations for Multitarget Applications
<code>download-mta-op-logs</code>	<code>dmol</code>	Download the log files for one or more operations concerning Multitarget Applications
<code>purge-mta-config</code>		Purge all configuration entries and subscriptions, which are no longer valid

### ⓘ Note

By default, large multitarget applications are not uploaded as a single unit, but are split up into smaller chunks of 45 MBs that are uploaded separately. Configure the chunk size as described in [Configuration](#).

## deploy

Deploy a new MTA or synchronize changes to an existing one. You have the following options for doing this:

### Usage

- **Deployment using a path to a directory**

You have the option to deploy or synchronize an MTA, the source of which is contained in a local directory:

```
cf deploy MTA_ARCHIVE | DIRECTORY_PATH [-e EXT_DESCRIPTOR[,...]] [-t TIMEOUT]
[--version-rule VERSION_RULE] [-u URL] [-f] [--retries RETRIES] [--no-start]
[--namespace NAMESPACE]
[--apply-namespace-app-names true/false] [--apply-namespace-service-names
true/false]
[--apply-namespace-app-routes true/false] [--apply-namespace-as-suffix true/
false ]
[--delete-services] [--delete-service-keys] [--delete-service-brokers] [--
keep-files]
[--no-restart-subscribed-apps] [--do-not-fail-on-missing-permissions] [--
abort-on-error]
[--strategy STRATEGY] [--skip-testing-phase] [--skip-idle-start] [--apps-
start-timeout TIMEOUT]
[--apps-stage-timeout TIMEOUT] [--apps-upload-timeout TIMEOUT] [--apps-task-
execution-timeout TIMEOUT]
```

- **Deployment using a URL to the MTA archive**

You have the option to deploy or synchronize an MTA, the source of which is contained at a URL address. When you use this command, the request prompts the backend to download the archive and then deploy it:

### ⚠ Caution

This option is currently experimental.

```
<write MTA archive URL to STDOUT> | cf deploy [-e EXT_DESCRIPTOR[,...]]  
[-t TIMEOUT] [--version-rule VERSION_RULE] [-u MTA_CONTROLLER_URL] [--retries  
RETRIES] [--no-start]  
[--namespace NAMESPACE] [--apply-namespace-app-names true/false] [--apply-  
namespace-service-names true/false]  
[--apply-namespace-app-routes true/false] [--apply-namespace-as-suffix true/  
false] [--delete-services]  
[--delete-service-keys] [--delete-service-brokers] [--keep-files] [--no-  
restart-subscribed-apps]  
[--do-not-fail-on-missing-permissions] [--abort-on-error] [--strategy  
STRATEGY] [--skip-testing-phase]  
[--skip-idle-start] [--apps-start-timeout TIMEOUT] [--apps-stage-timeout  
TIMEOUT]  
[--apps-upload-timeout TIMEOUT] [--apps-task-execution-timeout TIMEOUT]
```

### ⚠ Caution

There is no possibility for you to check if there are running ongoing operations on the MTA that you want to deploy using a URL.

### ⓘ Note

- The URL to the MTA archive must include the `<https://>` prefix.
- You can use an locally present extension descriptor along with this deployment method.

#### • Deployment from your current directory

You have the option to deploy the content of the directory you are currently in. The MultiApps CLI plugin packages the content in accordance with the deployment descriptor definition, so that the Cloud Deployment service can then upload and deploy the MTA.

```
cf deploy [-e <EXT_DESCRIPTOR_1>[,<EXT_DESCRIPTOR_2>]]  
[-u <URL>] [-t <TIMEOUT>] [-v <VERSION_RULE>]  
[--no-start] [--namespace]  
[--delete-services] [--delete-service-keys] [--delete-service-brokers]  
[] [--keep-files] [--no-restart-subscribed-apps] [--do-not-fail-on-missing-  
permissions] [--skip-idle-start]  
cf deploy ...[--retries <RETRIES>]
```

In addition to deployment, you can also interact with an active MTA deploy operation, for example, by performing an action:

```
cf deploy [-i <OPERATION_ID>] [-a <ACTION>]
```

## Arguments

The Cloud Deployment service uses the content of the mtad.yaml descriptor, and based its contained info assembles an MTA archive in that directory before deploying it.

## Command Arguments Overview

Argument	Description
<MTA_ARCHIVE>	The path to (and name of) the archive or the directory containing the Multitarget Application to deploy; the application archive must have the format (and file extension) <code>mtar</code> , for example, <code>MTApp1.mtar</code> . If the value of this argument is a directory, the checks if an <code>mtad.yaml</code> file exists in that directory, and based on it assembles an MTA archive before deploying it. If no argument exists, the current directory is used.
<DIRECTORY_PATH>	The Cloud Deployment service also accepts a path to a directory where an <code>mtad.yaml</code> is maintained with path elements, for example, <code>cf deploy ./</code> .
<--retries>	Retry the operation specified number of times in case a non-content error (default 3).

## Options

### Command Options Overview

Option	Description
<code>-e &lt;EXT_DESCRIPTOR_1&gt;[ , &lt;EXT_DESCRIPTOR_2&gt; ]</code>	Define one or more extensions to the deployment descriptors; multiple extension descriptors must be separated by commas.
<code>-u &lt;URL&gt;</code>	Specify the URL for the deployment-service end-point that is to be used for the deployment operation
<code>-t &lt;TIMEOUT&gt;</code>	Specify the maximum amount of time (in seconds) that the service must wait for before starting the deployed application
<code>-v &lt;VERSION_RULE&gt;</code>	Specify the rule to apply to determine how the application version number is used to trigger an application-update deployment operation, for example: "HIGHER", "SAME_HIGHER", or "ALL".
<code>-i &lt;OPERATION_ID&gt;</code>	Specify the ID of the deploy operation that you want to perform an action on
<code>-a &lt;ACTION&gt;</code>	Specify the action to perform on the deploy operation, for example, "abort", "retry", or "monitor", or "resume"
<code>-f</code>	Force deployment without requiring any confirmation about aborting any conflicting processes
<code>--no-start</code>	Do not start application after deployment
<code>--apps-upload-timeout &lt;TIMEOUT&gt;</code>	Defines how long, in seconds, you can upload your application binary before the MTA operation times out.  See <a href="#">Application-Specific Timeouts [page 636]</a> .

### ⓘ Note

This applies to all applications.

Option	Description
--apps-stage-timeout <TIMEOUT>	<p>Defines how long, in seconds, your application can take during staging before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>
	<p><b> ⓘ Note</b></p> <p>This applies to all applications.</p>
--apps-start-timeout <TIMEOUT>	<p>Defines how long, in seconds, your application can take to start before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>
	<p><b> ⓘ Note</b></p> <p>This applies to all applications.</p>
--apps-task-execution-timeout <TIMEOUT>	<p>Defines how long, in seconds, your application can take to execute a task before the MTA operation times out.</p> <p>See <a href="#">Application-Specific Timeouts [page 636]</a>.</p>
	<p><b> ⓘ Note</b></p> <p>This applies to all applications.</p>
--delete-services	Re-create changed services and delete discontinued services
--delete-service-keys	Delete the existing service keys and apply the new ones
--delete-service-brokers	Delete discontinued service brokers
--keep-files	Keep the files used for deployment
--no-restart-subscribed-apps	Do not restart subscribed applications that are updated during the deployment
--do-not-fail-on-missing-permissions	Perform the deployment, even if required administrator permissions are missing for some operations (for example, the creation of service brokers).
--abort-on-error	If an operation fails, the corresponding process is automatically aborted and cannot be retried using the option -a retry. However, if you run a new operation for the same MTA, you will not receive an error message that there is an ongoing process for the MTA and ask you if you want to abort it.
-m <module name>	Deploy only the module with the specified name.
	<p><b> ⓘ Note</b></p> <p>It can be used multiple times.</p>

Option	Description
--all-modules	<p>Deploy all modules.</p> <p><b> ⓘ Note</b></p> <p>Any <code>-m</code> options are ignored.</p>
-r <resource name>	<p>Deploy only the resource with the specified name</p> <p><b> ⓘ Note</b></p> <p>It can be used multiple times.</p>
--all-resources	<p>Deploy all resources.</p> <p><b> ⓘ Note</b></p> <p>Any <code>-r</code> options are ignored.</p>
--strategy	<p>Announce to the platform a special deployment approach, for example when performing a “blue-green” or an “incremental-blue-green” deployment. See <a href="#">Blue-Green Deployment Strategy [page 660]</a> and <a href="#">(Experimental) Incremental Blue-Green Deployment Strategy [page 663]</a>.</p> <p>Using this option, you will not be asked to manually confirm the deletion of the older version of the MTA applications. This means that the deployment process is performed without any interruptions and you are not prompted to confirm the switch of routes to the new version of the MTA applications.</p>
--skip-testing-phase	<p>When using the <code>--strategy</code> option for “blue-green” or “incremental-blue-green” deployment, you can choose to skip the testing phase and you will not be asked to manually confirm the deletion of the older version of the MTA applications. The deployment process is performed without any interruptions and you are not prompted to confirm the switch of routes to the new version of the MTA applications.</p> <p><b> ⓘ Note</b></p> <p>This option is equivalent to <code>--no-confirm</code> from “bg-deploy”.</p>
--namespace	<p>Namespace for the MTA. They are also applied to the application and service names. For more information, see <a href="#">(Experimental) Namespaces [page 666]</a>.</p> <p><b> ⓘ Note</b></p> <p>This option is currently experimental.</p>

Option	Description
--apply-namespace-app-names true/false	Apply namespace to application names. If the namespace value is not provided in the CLI options, it is not applied.
<b> ⓘ Note</b> This option is currently experimental.	<b> ⓘ Note</b> This is applied on all applications.
--apply-namespace-service-names true/false	Apply namespace to service names. If the namespace value is not provided in the CLI options, it is not applied.
<b> ⓘ Note</b> This option is currently experimental.	<b> ⓘ Note</b> This is applied on all services.
--apply-namespace-app-routes true/false	Apply namespace to application routes. If the namespace value is not provided in the CLI options, it is not applied.
<b> ⓘ Note</b> This option is currently experimental.	<b> ⓘ Note</b> This is applied on all application routes.
--apply-namespace-as-suffix true/false	Apply namespace as suffix. If the namespace value is not provided in the CLI options, it is not applied.
<b> ⓘ Note</b> This option is currently experimental.	<b> ⓘ Note</b> This is applied on all applications, services, and application routes.
--skip-idle-start	When using the --strategy option for “blue-green”, you can choose to skip the starting of the newly deployed applications on idle routes. This means that the newly deployed applications will be mapped directly to live routes. Under the hood this option includes --skip-testing-phase.
<b> ⓘ Note</b> This option is currently experimental.	<b> ⓘ Note</b> <ul style="list-style-type: none"> <li>• If any of the options -m, --all-modules, -r, --all-resource is used, only the specified modules and resources will be deployed. Otherwise, everything will be deployed by default.</li> <li>• If the options for the module deploying (-m, --all-modules) are used, the modules need to contain a path element on an MTA module level, which points to the content of the module. In case the module has some requires dependency section to a resource that needs a configuration file, then the requires section should have a path parameter, which points to the configuration file.</li> <li>• If the options for the resource deploying (-r, --all-resource) are used, then any resources that have some configuration files need to contain a path parameter in their parameters section, which points to the configuration file.</li> <li>• The path element or parameter value should be relative to the MTA directory.</li> </ul>

An example of an MTA deployment descriptor, containing all variants of the `path` elements and parameters can be found at [Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 578\]](#).

## bg-deploy

Deploy a new Multitarget Application (MTA) using “blue-green” (zero-downtime) deployment.

### ⓘ Note

You can also perform this deployment type using the `deploy` command by using the experimental `--strategy blue-green` flag. See above for details.

“Blue-green” deployment is a release technique that reduces application downtime and the resulting risk by running two identical target deployment environments called “blue” and “green”. Only one of the two target environments is “live” at any point in time and it is much easier to roll back to a previous version after a failed (or undesired) deployment.

### Usage

Deploy a new Multitarget Application using “blue-green” deployment:

```
cf bg-deploy <MTA_ARCHIVE>
[-e <EXT_DESCRIPTOR_1>[,<EXT_DESCRIPTOR_2>]]
[-u <URL>] [-t <TIMEOUT>] [-v <VERSION_RULE>]
[--no-start] [--use-namespaces] [--no-namespaces-for-services]
[--delete-services] [--delete-service-keys] [--delete-service-brokers]
[--keep-files] [--no-restart-subscribed-apps] [--no-confirm] [--do-not-fail-on-
missing-permissions][--skip-idle-start]
cf deploy ...[--retries <RETRIES>]
```

Interact with an active MTA deploy operation, for example, by performing an action:

```
cf bg-deploy [-i <OPERATION_ID>] [-a <ACTION>]
```

### Arguments

Command Arguments Overview

Argument	Description
<MTA_ARCHIVE>	The path to (and name of) the archive or the path to the directory containing the Multitarget Application to deploy. The application archive must have the format (and file extension) <code>.mtar</code> , for example, <code>MTApp1.mtar</code> ; the specified directory can be specified as a path (for example, <code>myApp/</code> or <code>.</code> (current directory)).
<code>--retries</code>	Retry the operation specified number of times in case a non-content error (default 3).

## Options

Command Options Overview

Option	Description
-e <EXT_DESCRIPTOR_1>[ , <EXT_DESCRIPTOR_2>]	Define one or more extensions to the deployment descriptors; multiple extension descriptors must be separated by commas.
-u <URL>	Specify the URL for the deployment-service end-point that is to be used for the deployment operation
-t <TIMEOUT>	Specify the maximum amount of time (in seconds) that the service must wait for before starting the deployed application
-v <VERSION_RULE>	Specify the rule to apply to determine how the application version number is used to trigger an application-update deployment operation, for example: "HIGHER", "SAME_HIGHER", or "ALL".
-i <OPERATION_ID>	Specify the ID of the deploy operation that you want to perform an action on
-a <ACTION>	Specify the action to perform on the deploy operation, for example, "abort", "retry", or "monitor", or "resume"
-f	Force deploy without requiring any confirmation for aborting any conflicting processes
--no-start	Do not start application after deployment
--use-namespaces	Use namespaces in application (and service) names during application deployment
--no-namespaces-for-services	Do not use namespaces in service names
--delete-services	Re-create changed services and delete discontinued services
--delete-service-keys	Delete the existing service keys and apply the new ones
--delete-service-brokers	Delete discontinued service brokers
--keep-files	Keep the files used for deployment
--no-restart-subscribed-apps	Do not restart subscribed applications that are updated during the deployment
--no-confirm	Use this option to turn off the manual confirmation for deleting the older version of the MTA applications. Thus the deployment process is performed from start to finish uninterrupted, and you are not prompted to confirm the switch of routes to the new version of the MTA applications.
--do-not-fail-on-missing-permissions	Perform the deployment, even if required administrator permissions are missing for some operations (for example, the creation of service brokers).
--abort-on-error	If an operation fails, the corresponding process is automatically aborted and cannot be retried using the option -a retry

Option	Description
<code>-m &lt;module name&gt;</code> . However, if you run a new operation for the same MTA, you will not receive an error message that there is an ongoing process for the MTA and ask you if you want to abort it.	<p>Deploy only the module with the specified name.</p> <p><b> ⓘ Note</b></p> <p>It can be used multiple times.</p>
<code>--all-modules</code>	<p>Deploy all modules.</p> <p><b> ⓘ Note</b></p> <p>Any <code>-m</code> options are ignored.</p>
<code>-r &lt;resource name&gt;</code>	<p>Deploy only the resource with the specified name</p> <p><b> ⓘ Note</b></p> <p>It can be used multiple times.</p>
<code>--all-resources</code>	<p>Deploy all resources.</p> <p><b> ⓘ Note</b></p> <p>Any <code>-r</code> options are ignored.</p>
<code>--skip-idle-start</code>	<p>You can choose to skip the starting of the newly deployed applications on idle routes. This means that the newly deployed applications will be mapped directly to live routes. Under the hood this option includes <code>--no-confirm</code>.</p> <p><b> ⓘ Note</b></p> <p>This option is currently experimental.</p>

### ⓘ Note

- If any of the options `-m`, `--all-modules`, `-r`, `--all-resource` is used, only the specified modules and resources will be deployed. Otherwise, everything will be deployed by default.
- If the options for the module deploying (`-m`, `--all-modules`) are used, the modules need to contain a `path` element on an MTA module level, which points to the content of the module. In case the module has some `requires` dependency section to a resource that needs a configuration file, then the `requires` section should have a `path` parameter, which points to the configuration file.
- If the options for the resource deploying (`-r`, `--all-resource`) are used, then any resources that have some configuration files need to contain a `path` parameter in their parameters section, which points to the configuration file.
- The `path` element or parameter value should be relative to the MTA directory.

See the complete procedure in section [Legacy Blue-Green Deployment \[page 658\]](#).

An example of an MTA deployment descriptor, containing all variants of the `path` elements and parameters can be found at [Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 578\]](#).

# undeploy

Undeploy a Multitarget Application (MTA), or interact with an undeploy MTA operation.

## Usage

Undeploy an MTA.

```
cf undeploy <MTA_ID>
[-u <URL>] [-f]
[--delete-services] [--delete-service-keys] [--delete-service-brokers] [--no-
restart-subscribed-apps]
[--do-not-fail-on-missing-permissions]
cf deploy ...[--retries <RETRIES>]
```

Interact with an undeploy MTA operation, for example, by performing an action.

```
cf undeploy [-i <UNDEPLOY_ID>] [-a <ACTION>]
```

## Arguments

Command Arguments Overview

Argument	Description
<MTA_ID>	The ID of the MTA you want to undeploy
<--retries>	Retry the operation specified number of times in case a non-content error (default 3).

## Options

Command Options Overview

Option	Description
-u <URL>	Specify the URL for the service end-point that is to be used for the undeployment operation
-i <OPERATION_ID>	Specify the ID of the undeploy operation that you want to perform an action on
-a <ACTION>	Specify the action to perform on the undeploy operation, for example, "abort", "retry", or "monitor"
-f	Force completion of the undeploy operation without any system prompt or confirmation
--delete-services	Delete any related services
--delete-service-brokers	Delete discontinued service brokers
--no-restart-subscribed-apps	Do not restart subscribed applications that are updated during the deployment
--do-not-fail-on-missing-permissions	Perform the deployment, even if required administrator permissions are missing for some operations (for example, the creation of service brokers).

Option	Description
--abort-on-error	If an operation fails, the corresponding process is automatically aborted and cannot be retried using the option -a retry. However, if you run a new operation for the same MTA, you will not receive an error message that there is an ongoing process for the MTA and ask you if you want to abort it.
--delete-service-keys	Delete the existing service keys

## mta

Display information about a Multitarget Application (MTA). The information displayed includes the requested state, the number of instances, information about allocated memory and disk space, as well as details regarding the bound service (and service plan).

### Usage

```
cf mta MTA_ID
[-u <URL>]
```

### Arguments

Command Arguments Overview

Argument	Description
<MTA_ID>	The ID of the MTA whose details you want to display

### Options

Command Options Overview

Option	Description
-u <URL>	Specify the URL for the deployment-service end-point to use to obtain details of the selected MTA

## mtas

Display information about all available Multitarget Applications (MTA).

### Usage

```
cf mtas [-u <URL>]
```

## Options

Command Options Overview

Option	Description
-u <URL>	Specify the URL for the deployment-service end-point to use to obtain details of the selected MTA

## mta-ops

Display information about all **active** operations for Multitarget Applications (MTA). The information includes the ID, type, status, the time the MTA-related operation started, as well as the name of the user that started the operation.

### Usage

```
cf mta-ops [--mta <MTA>] [-u <URL>] [--last <NUM>] [--all]
```

## Options

Command Options Overview

Option	Description
-u <URL>	Specify the URL for the deployment-service end-point to use to obtain details of the selected MTA operations
--last <NUM>	List the last <NUM> active MTA operations
--all	List all active MTA operations
- mta <MTA>	Specify the MTA whose operations you want to list.

## download-mta-op-logs

Download the log files for one or more operations concerning Multitarget Applications.

### Usage

You have the following usage options:

- `cf download-mta-op-logs -i <OPERATION_ID> [-u <URL>] [-d <DIRECTORY>]`
- `cf download-mta-op-logs --mta <MTA> [ --last <NUM> ] [ -u <URL> ] [ -d <DIRECTORY> ]`

### → Tip

You can use the alias `dmo1` in place of the `download-mta-op-logs` command.

## Options

Command Options Overview

Option	Description
-u <URL>	Specify the URL for the deployment-service end-point to use to obtain details of the selected MTA operations
-i <OPERATION_ID>	Specify the identity (ID) of the MTA operation whose logs you want to download
-d <DIRECTORY>	Specify the path to the location where you want to save the downloaded MTA operation logs; by default, the location is . /mta-op-<OPERATION_ID> /
--mta <MTA>	Specify the MTA whose logs you want to download.
--last<NUM>	Download the last <NUM> logs for the specified MTA.

### ⓘ Note

This option is not compatible with the -i option.

## purge-mta-config

Purge all configuration entries and subscriptions, which are no longer valid.

### Usage

```
cf purge-mta-config  
[-u <URL>]
```

Invalid configuration entries are often produced when the application that is providing configuration entries is deleted by the user without using the deploy-service, for example, the cf delete command. In this case, the configuration remains in the deploy-service database even though the corresponding application is no longer available. This could lead to a failure during subsequent attempts to resolve the configuration entries.

## Options

Command Options Overview

Option	Description
-u <URL>	The URL of the deploy service

## Related Information

[MultiApps CLI plugin](#)

[Cloud Foundry multitarget application examples repository](#)

[Deploy an MTA archive referenced by a remote URL](#)

## 6.2.5.1.3.1.2.1 Application Versions and Deployment Consistency

The SAP Cloud Deployment service follows the rules specified in the Semantic Versioning Specification (Semver) when comparing the versions of a deployed multitarget application (MTA) with the version that is to be deployed. If an MTA that is submitted for deployment has a version that is lower than or equal to an already deployed version of the same MTA, the deployment might fail if there is a conflict with the version rule specified in the deployment operation. The version rule is a parameter of the deployment process, which specifies which relationships to consider between the existing and the new version of an MTA before proceeding with the operation. The version rules supported by the deploy service are as follows:

- HIGHER - Only MTAs with versions that are bigger than the version of the currently deployed MTA, are accepted for deployment.
- SAME\_HIGHER - Only MTAs with versions that are higher than (or the same as) the version of the currently deployed MTA are accepted for deployment. This is the **default** setting for the version rule.
- ALL - All versions are accepted for deployment.

## 6.2.5.1.3.2 Service Fabrik Plugin in Cloud Foundry

Service Fabrik CF CLI plugin is used for performing various backup and restore operations on service-instances in Cloud Foundry, such as starting/aborting a backup, listing all backups, removing backups, starting/aborting a restore, etc.

This CF CLI plugin is only available for ServiceFabrik broker, so it can only be used with CF installations in which this service broker is available. You can also list all available commands and their usage with 'cf backup'. You can now manage your backups of service instance and restore them using the backup and restore functionality.

To use the functionality, use the SAP BTP cockpit or the extended Cloud Foundry commands in the command line interface.

### 6.2.5.1.3.2.1 Install the Service Fabrik Plugin

The Service Fabrik plugin lets you manage backups of a service instance by extending Cloud Foundry commands.

#### Prerequisites

You need to have Cloud Foundry Command Line Interface (CF CLI) installed for the plugin to work since it is built on CF CLI. The installation instructions for CF CLI can be found [here](#). The minimum version of CF CLI, on which the plugin has been tested successfully is v6.20.

## Procedure

1. Download the latest version of the plugin that is compatible with your operating system. On the Web page, you will find the plugin under the **SAP Cloud Platform Cloud Foundry CLI Plugins** section with the name **Service Fabrik based B&R**.
2. Untar or unzip the downloaded archive.
3. Open the command line interface or terminal.
4. To install the plugin, enter the following command:
  - Mac and Linux: `cf install-plugin <path to the downloaded folder></service-fabrik-cli-plugin [Linux/Mac]>`
  - Windows: `cf install-plugin<path to the downloaded folder><\service-fabrik-cli-plugin.exe [Windows]>`

### ⓘ Note

If you are reinstalling the plugin, first uninstall the previous version using: `cf uninstall-plugin ServiceFabrikPlugin`

5. Verify that the plugin has been installed successfully by entering `cf plugins`.

You see a list of plugins that now includes Service Fabrik. The output also displays commands that are specific to the Service Fabrik plugin.

## Related Information

<https://tools.hana.ondemand.com/#cloud>

[Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

## 6.2.5.1.3.2.2 Extended Cloud Foundry Commands of Service Fabrik

The Service Fabrik plugin provides commands that support backup and restore operations.

The various commands described in the table below are functionalities that facilitate these operations:

Usage	Description	Common Error Scenarios
<code>cf list-backup</code>	Shows a list of all the service instance backups. These backups are specific to the space you are logged on to. If you have permission to access multiple spaces and need to view backups for a specific space, log on to that space in Cloud Foundry.	<b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.
<code>cf list-backup &lt;service_instance_name&gt;</code>	Shows a list of backups that are specific to the service instance within a space.	<b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.
<code>cf backup &lt;backup_id&gt;</code>	Provides additional information about a specific backup such as the service instance name for which backup was taken, the org and the space name to which the backup belongs, the type of backup, the status of backup, the start and finish time of backup, and so on.	none
<code>cf list-backup --guid &lt;service_instance_guid&gt;</code>	Shows the list of all backups for the given service-fabrik service instance. The argument has to be the guid of the service instance. (Works even for a deleted instance.)	<b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.
<code>cf list-backup &lt;service_instance_name&gt; --deleted</code>	Shows the list of all backups for a deleted service-fabrik service instance. (Works only for a deleted service-instance.)	<b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission. <b>Deleted instance not found error:</b> If the deleted instance name is incorrect.
		<b>Multiple Guid for a deleted service instance error:</b> When plugin encounters multiple instance Guid's associated with the given instance name

Usage	Description	Common Error Scenarios
<pre>cf start-restore &lt;service_instance_name&gt; &lt;backup_id&gt;</pre>	<p>Restores a service instance from the specified instance name and backup ID. Before providing a restore command, ensure that the backup is available for the service instance. You can verify the state of the restore process using <code>cf restore &lt;service_instance_name&gt;</code>.</p>	<p><b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.</p> <p><b>Another concurrent access:</b> if another operation is already in progress for the service instance. You might need to try again after the current operation is completed.</p>
<pre>cf abort-restore &lt;service_instance_name&gt;</pre>	<p>Aborts an ongoing restore operation of the specified service instance. You can verify the state of the abort process using <code>cf restore &lt;service_instance_name&gt;</code>.</p>	<p><b>No restore in progress:</b> if no restore operation is currently in progress for the specified service instance. The restore activity might have completed or you might not have initiated a restore.</p> <p><b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.</p>
<pre>cf instance-events --delete</pre>	<p>List all delete service instance events in the space.</p>	<p><b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.</p>
<pre>cf restore&lt;service_instance_name&gt;</pre>	<p>Check the progress of the last restore operation triggered on the specified instance name. Provides additional information about the restore operation such as the service instance name for which restore was taken, the org and the space name to which the instance belongs, the status of restore, the start and finish time of restore, and so on.</p>	<p><b>No restore in progress:</b> if no restore operation is currently in progress for the specified service instance. The restore activity might have completed or you might not have initiated a restore.</p> <p><b>Unauthorized Access:</b> if you do not have permission to access the space containing the service instance or the service instance itself. Verify that you have the required permission.</p>

### 6.2.5.1.3.3 Custom Domain Plugin for the Cloud Foundry Environment

The Custom Domain CLI plugin provides functions for creating private keys and certificate signing requests, as well as additional commands for managing your custom domains.

To use the functionality of the Custom Domain plugin, use the extended Cloud Foundry commands in the command-line interface.

#### Related Information

[Install the Custom Domain Plugin \[page 2474\]](#)

[Extended Cloud Foundry Commands of Custom Domains \[page 2476\]](#)

[Configuring Custom Domains](#)

#### 6.2.5.1.3.3.1 Install the Custom Domain Plugin

Use the Custom Domain CLI plugin to configure and manage your custom domains.

#### Prerequisites

Install the Cloud Foundry command-line interface (CLI) for the plugin to work. You can find the installation instructions here: [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#).

#### Procedure

1. Download the latest version of the plugin that is compatible with your operating system. On the [Web page](#), you'll find the plugin under the **SAP BTP Cloud Foundry CLI plugins** section with the name Custom Domain.
2. Untar or unzip the downloaded archive.
3. Open the command-line interface or terminal.
4. To install the plugin, enter the following command:

- Mac and Linux:

```
cf install-plugin <path to the downloaded folder>/custom-domain-cli
```

- Windows:

```
cf install-plugin <path to the downloaded folder>\custom-domain-cli.exe
```

### Note

If you are reinstalling the plugin, first uninstall the previous version using: `cf uninstall-plugin "Custom Domain"`

5. Verify that the plugin has been installed successfully by entering:

```
cf plugins
```

You see a list of plugins that now includes Custom Domain. The output also displays commands that are specific to the Custom Domain plugin.

## Results

You have installed the Custom Domain plugin for the Cloud Foundry CLI and can now use the extended commands that are available from the Custom Domain service.

## Related Information

[Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#)

[Configuring Custom Domains](#)

## 6.2.5.1.3.3.2 Extended Cloud Foundry Commands of Custom Domains

The Custom Domain plugin includes commands that you can use to configure and manage your custom domains.

Commands for the Custom Domain CLI Plugin

Command	Alias	Usage	Options	Description
cf custom-domain-create-key	cdck	cf custom-domain-create-key KEY SUBJECT DOMAIN [DOMAIN ...] [options]	-ski p-ssl-vali dati on -ver bose -for ce	Do not attempt to validate SSL certificate. Generate a new private and public key pair. Show verbose information. Force key creation without confirmation.
cf custom-domain-get-csr	cdgc	cf custom-domain-get-csr KEY [FILE] [options]	-ski p-ssl-vali dati on -ver bose	Download the certificate signing request corresponding to a new key. Show verbose information.

Command	Alias	Usage	Options	Description
cf custom-domain-upload-certificate-chain	cducc	cf custom-domain-upload-certificate-chain KEY FILE [options]	-ski p- ssl- vali dati on	Do not attempt to validate SSL certificate.  -ver bose Show verbose information.
			-for ce	Force uploading the certificate chain without confirmation.
cf custom-domain-activate	cda	cf custom-domain-activate KEY DOMAIN [DOMAIN ...] [options]	-ski p- ssl- vali dati on	Activate the server certificate for specified domains.  -ver bose Show verbose information.

Command	Alias	Usage	Options	Description
cf custom-domain-list	cdl	cf custom-domain-list	<p>-ski    Do not attempt to validate SSL certificate.</p> <hr/> <p>-ver    Show verbose information.</p>	List domains and their configuration status.
cf custom-domain-show-certificates	cdsc	cf custom-domain-show-certificates KEY [FILE] [options]	<p>-dump    Dump the certificates in PEM format.</p> <hr/> <p>-ski    Do not attempt to validate SSL certificate.</p> <hr/> <p>-verbose    Show verbose information.</p>	Show certificates for a specified key.

Command	Alias	Usage	Options	Description
<code>cf custom-domain-enable-client-authentication</code>	cdeca	<code>cf custom-domain-enable-client-authentication [FILE DOMAIN ...] [options]</code>	<p>-ski p- ssl- vali dati on</p> <p><b>-ver</b> <b>bose</b></p>	Activate client authentication for specified domains.  Do not attempt to validate SSL certificate.  Show verbose information.
<code>cf custom-domain-disable-client-authentication</code>	cdcca	<code>cf custom-domain-disable-client-authentication DOMAIN [options]</code>	<p>-ski p- ssl- vali dati on</p> <p><b>-ver</b> <b>bose</b></p> <p><b>-force</b></p>	Disable client authentication for a domain.  Do not attempt to validate SSL certificate.  Show verbose information.  Force deactivation of client authentication without confirmation.

Command	Alias	Usage	Options	Description
cf custom-domain-show-trusted-certificates	cdstc	cf custom-domain-show-trusted-certificates DOMAIN [FILE] [options]	<p>-dum p</p> <p>-ski p-ssl-vali dati on</p> <p>-ver bose</p>	Show the trusted certificates for a domain.
cf custom-domain-deactivate	cdd	cf custom-domain-deactivate DOMAIN [options]	<p>-ski p-ssl-vali dati on</p> <p>-ver bose</p> <p>-for ce</p>	Deactivate the certificate for a specified domain.

Command	Alias	Usage	Options	Description
cf custom-domain-delete-key	cddk	cf custom-domain-delete-key KEY [options]	<p>-ski      Do not p-            at- ssl-        tempt vali        to vali- dati        date on            SSL               certifi-               cate.</p> <hr/> <p>-ver      Show bose     ver-           bose           infor-           mat-           ion.</p> <hr/> <p>-for      Force ce           key           dele-           tion           with-           out           confir-           mat-           ion.</p>	Delete the private key and its certificates.
cf custom-domain-version	cdv	cf custom-domain-version	<p>-ski      Do not p-            at- ssl-        tempt vali        to vali- dati        date on            SSL               certifi-               cate.</p> <hr/> <p>-ver      Show bose     ver-           bose           infor-           mat-           ion.</p>	Show the Custom Domain plugin version.

Command	Alias	Usage	Options	Description
cf custom-domain-map-route	cdmr	cf custom-domain-map-route <b>STANDARD_ROUTE</b>   <b>CUSTOM_ROUTE</b>	-skip-attempt -p-ssl-validation -validate-ssl-certificate  -verbose	Do not attempt to validate SSL certificate.  Show verbose information.
cf custom-domain-unmap-route	cdur	cf custom-domain-unmap-route <b>CUSTOM_ROUTE</b>	-skip-attempt -p-ssl-validation -validate-ssl-certificate  -verbose	Do not attempt to validate SSL certificate.  Show verbose information.
cf custom-domain-routes	cdr	cf custom-domain-routes	-skip-attempt -p-ssl-validation -validate-ssl-certificate  -verbose	List all configured routes of SaaS applications with custom domains.

## Related Information

[Configuring Custom Domains](#)

[Managing Custom Domains](#)

### 6.2.5.2 Add Organization Members Using the Cloud Foundry Command Line Interface

You can use the Cloud Foundry Command Line Interface (cf CLI) to add organization members and assign roles to them.

#### Prerequisites

- The Cloud Foundry environment is enabled.  
Enable it in the SAP BTP cockpit. For more information, see [Create a Subaccount \[page 2173\]](#).
- The Org Manager role for the org in question.

#### ⓘ Note

You automatically have the Org Manager role in an org that you created.

- The user exists in the UAA.

All users are stored in identity providers, either in the default or in a custom identity provider. Cloud Foundry needs a copy of the user, sometimes called a shadow user. To create the shadow user, choose one of the following options:

- The user has been added from the cockpit.  
For more information, see [Add Org Members \[page 2437\]](#).
- The user has already logged on to the Cloud Foundry environment.  
For example: `cf login -a https://api.cf.eu20.hana.ondemand.com`

#### ⓘ Note

The landscape the user logs on to must be the same as the landscape where the subaccount is located. See the overview page of your subaccount for the API endpoint.

- You have the e-mail addresses of the members that you want to add.

The user must already exist in the connected identity provider. For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

## Context

This task is meant for the `OrgManager` and `OrgAuditor` roles. To add members with the `OrgUser` role, follow the procedure described in [Add Space Members Using the Cloud Foundry Command Line Interface \[page 2486\]](#).

## Procedure

Enter the following command, specifying the user name, the name of the organization, and the role:

```
cf set-org-role <USERNAME> <ORG> <ROLE>
```

```
cf set-org-role julie.armstrong@example.com account_subaccount-47v2w8dj  
OrgAuditor
```

## Next Steps

To remove an org role from a user, enter the following string, specifying the user name, the name of the organization, and the role:

```
cf unset-org-role <USERNAME> <ORG> <ROLE>
```

## Related Information

[Supported Tools and Services When Using Custom Identity Providers for Platform Users \[page 2248\]](#)

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

[Managing Org Members Using the Cockpit \[page 2436\]](#)

## 6.2.5.3 Create Spaces Using the Cloud Foundry Command Line Interface

Use the `cf create-space` command to create spaces in your Cloud Foundry organization using the Cloud Foundry Command Line Interface (cf CLI).

### Prerequisites

- (Enterprise accounts only) Create at least one subaccount and enable the Cloud Foundry environment in this subaccount. For more information, see [Create a Subaccount \[page 2173\]](#) and [Create Orgs \[page 2435\]](#).

#### ⓘ Note

In a trial account, the Cloud Foundry environment is automatically activated, and a first space named `dev` is created.

- You have the Org Manager role in the organization in which you want to create a space. For more information about roles and permissions, see <https://docs.cloudfoundry.org/concepts/roles.html>.
- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

### Procedure

1. Make sure that you are in the Cloud Foundry organization you want to add a space to.

```
cf target -o ORG
```

To get a list of your organizations, call `cf orgs`.

2. Enter the following string, specifying a name for the new space, the name of the organization, and the quota you'd like to assign to it:

```
cf create-space SPACE [-o ORG] [-q SPACE_QUOTA]
```

3. (Optional) Set this space as target by executing: `cf target -o ORG -s SPACE`

#### ⓘ Note

If you are assigned to only one Cloud Foundry organization and space, the system automatically targets you to the relevant Cloud Foundry organization and space when you log on.

## Related Information

[Delete Spaces Using the Cloud Foundry Command Line Interface \[page 2490\]](#)

[Create Spaces Using the Cockpit \[page 2441\]](#)

### 6.2.5.4 Add Space Members Using the Cloud Foundry Command Line Interface

You can use the Cloud Foundry Command Line Interface (cf CLI) to add space members and assign roles to them.

#### Prerequisites

- The Space Manager role for the space.
- The users exist in a trusted platform identity provider.

All users of Cloud Foundry UAA are stored in identity providers, either in the default or in a custom identity provider. UAA needs a copy of the user, sometimes called a shadow user. You assign the shadow user authorizations to access resources in UAA. When a user authenticates, UAA forwards the request to the identity provider.

To specify which identity provider hosts the user, use the `--origin` option. For more information about finding the origin key, see the trust configuration documentation.

For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

#### Context

Space members are organization members who have specific roles in a space.

Org members can only be added by an Org Manager. This means that if you only have the Space Manager role, you can't add space members that aren't known to the org. To do that, you must first ask your Org Manager to add the users as org members with no roles. Once this is done, you can add them as space members following the steps below.

If you're the Org Manager, you don't need to first add the users as org members with no roles. Since you have the permissions necessary to add users to the org, when you add a new user as a space member, that user automatically becomes part of the org as well.

## Procedure

Enter the following string, specifying the user name, the name of the organization, the name of the space, and the role:

```
cf set-space-role USERNAME ORG SPACE ROLE
```

## Next Steps

To remove a space role from a user, enter the following string, specifying the user name, the name of the organization, the name of the space, and the role:

```
cf unset-space-role USERNAME ORG SPACE ROLE
```

## Related Information

[Supported Tools and Services When Using Custom Identity Providers for Platform Users \[page 2248\]](#)

[Add Space Members Using the Cockpit \[page 2444\]](#)

## 6.2.5.5 Create Space Quota Plans Using the Cloud Foundry Command Line Interface

You can use the Cloud Foundry Command Line Interface to create space quota plans.

### Prerequisites

- The Org Manager role for the org that contains the spaces to which you want to assign quotas.
- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

## Procedure

Open a command line and enter the following string, replacing QUOTA with the name for your space quota plan:

```
cf create-space-quota QUOTA [-i INSTANCE_MEMORY] [-m MEMORY] [-r ROUTES] [-s SERVICE_INSTANCES] [--allow-paid-service-plan | --disallow-paid-service-plans]
```

Specify the following parameters:

- `-i`: Maximum amount of memory an application instance can have (`-1` represents an unlimited amount)
- `-m`: Total amount of memory a space can have
- `-r`: Total number of routes
- `-s`: Total number of service instances
- `--allow-paid-service-plans`: Can provision instances of paid service plans
- `--disallow-paid-service-plans`: Can not provision instances of paid service plans

## Related Information

[Create Space Quotas \[page 2449\]](#)

### 6.2.5.6 Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface

You use the Cloud Foundry Command Line Interface to assign the quotas available in your global account to your subaccounts.

#### Prerequisites

- The Org Manager role for the org that contains the spaces to which you want to assign quotas.
- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
- A space quota plan. For more information, see [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 2487\]](#).

#### Procedure

Open a command line and enter the following string:

```
cf set-space-quota SPACE_NAME SPACE_QUOTA_NAME
```

## 6.2.5.7 Change Space Quota Plans Using the Command Line Interface

Change space quota plans in the Cloud Foundry environment using the Cloud Foundry command line interface (cf CLI).

### Prerequisites

- You have the Org Manager role in the organization in which you'd like to change space quota plans. For more information about roles and permissions, see <https://docs.cloudfoundry.org/concepts/roles.html>.
- Download and install the cf CLI and log on to your Cloud Foundry instance. For more information, see [Download and Install Cloud Foundry Command Line Interface](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

### Procedure

1. (Optional) Enter the following string to identify the names of all space quota plans available in your org:

```
cf space-quotas
```

2. To modify a quota plan, enter the following string:

```
cf update-space-quota SPACE-QUOTA-NAME [-i MAX-INSTANCE-MEMORY] [-m MEMORY]
[-n NEW_NAME] [-r ROUTES] [-s SERVICES] [--allow-paid-service-plans | --
disallow-paid-service-plans]
```

#### ⓘ Note

For more information about managing space quota plans, see <https://docs.cloudfoundry.org/adminguide/quota-plans.html#space>.

### Related Information

[Edit Space Quotas \[page 2451\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

## 6.2.5.8 Delete Spaces Using the Cloud Foundry Command Line Interface

Use the `cf delete-space` command to delete spaces in your Cloud Foundry organization using the Cloud Foundry Command Line Interface (cf CLI).

### Prerequisites

- You have the Org Manager role in the organization from which you want to delete a space. For more information about roles and permissions, see <https://docs.cloudfoundry.org/concepts/roles.html>.
- Download and install the cf CLI and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 2452\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).

### Procedure

1. Make sure that you are in the Cloud Foundry organization you want to delete a space in.

```
cf target -o ORG
```

To get a list of your organizations, call `cf orgs`.

2. Enter the following string, specifying a name for the space, and the name of the organization:

```
cf delete-space SPACE [-o ORG] [-f]
```

To get the list of your spaces, call `cf spaces`.

### Related Information

[Create Spaces Using the Cloud Foundry Command Line Interface \[page 2485\]](#)

[Delete Spaces Using the Cockpit \[page 2443\]](#)

## 6.2.6 Application Operations in the Cloud Foundry Environment

Learn more about the different application operations that you can perform in the Cloud Foundry environment.

- [Deploy an Application \[page 2491\]](#)
- [Start, Stop and Restart Applications \[page 2492\]](#)

- Add or Remove Application Instances [page 2494]
- Configuring Application URLs in the Cloud Foundry CLI [page 2494]
  - About Routes in the Cockpit [page 2495]
  - Create Routes [page 2496]
  - Map Routes to Applications [page 2498]
  - Using Custom Domains [page 2501]

### 6.2.6.1 Deploy an Application

You can use the cockpit to deploy a new application in the Cloud Foundry environment.

#### Procedure

1. In your Cloud Foundry subaccount, navigate to the space where you would like to deploy your application.
2. On the *Applications* page, choose *Deploy Application*.
3. Choose the location of the file which contains your application.
4. (Optional) If you would like to use a manifest, choose the location of your `manifest.yml` file.

#### ⓘ Note

To avoid any unexpected behavior, specify the attributes of your choice and their respective values in the `manifest.yml` file. For reference, see <https://docs.cloudfoundry.org/devguide/deploy-apps/manifest-attributes.html>.

5. (Optional) If you don't want to use a manifest, deselect the *Use Manifest* box and enter the application details.
  - a. Enter a name for your application.
  - b. (Optional) Edit the amount of memory and disk space available to each instance of your app, as well as the number of instances.

The amount of memory per instance that you assign, in this step or in the `manifest.yml` file during deployment, is the value that is metered when using the Cloud Foundry runtime service. For more information, see the examples in [Monitoring and Troubleshooting](#).

#### ⓘ Note

By default, each instance of a new app is assigned 1024 MB of memory and 512 MB of disk space, and each app starts with 1 instance. If you require more or less resources, edit the prefilled fields in the form to suit your needs.

- c. (Optional) If you don't need a route for your app, select the *No Route* box.
- d. (Optional) If you would like to create a route for your app, leave the *No Route* box deselected and choose a host name (if different than your app name) and a domain for your app.

### Note

When you enter an app name, the *Host* field is automatically filled with the same name. You can make changes to it or leave it as is. After deciding on a host and domain, you can see a preview of your final application route at the bottom of the form.

6. Choose *Deploy*.

## Results

The file containing your new application is uploaded and your application is deployed.

## Related Information

[About Routes in the Cockpit \[page 2495\]](#)

[Create Routes \[page 2496\]](#)

### 6.2.6.2 Start, Stop and Restart Applications

You can start and stop applications in the Cloud Foundry environment to control whether they can be accessed by end users.

## Prerequisites

You have deployed an app in your Cloud Foundry space.

## Context

A user can only reach your application if the application is started. The application routes do not work when the application is stopped.

The first start of the application occurs when you deploy the app, if enough quota is available in the space.

## Procedure

1. Navigate to the [Applications](#) page in your space.
2. You can perform these actions either directly from the [Applications](#) page, or by navigating into your application, to the [Overview](#) page:

Page	Actions
<a href="#">Applications page</a>	<p>For the application that you'd like to start or stop, choose the respective icon from the <a href="#">Actions</a> column:</p> <ul style="list-style-type: none"><li>•  (<i>Start</i>) - This starts the first instance of your application and makes it available to users.</li><li>•  (<i>Stop</i>) - This stops all running instances of the application.</li></ul> <p>You cannot directly restart an application from this page.</p>
<a href="#">Overview page of your application</a>	<p>Choose the button corresponding to the desired action:</p> <ul style="list-style-type: none"><li>• <i>Start</i> - This starts the first instance of your application and makes it available to users.</li><li>• <i>Stop</i> - This stops all running instances of the application.</li><li>• <i>Restart</i> - This restarts the application.</li></ul> <p> <b>Note</b> When you start or stop an application from this page, the state shown in the table changes. However, this is not the actual state of the application, but the state that has been requested through your action. To check the actual state of the application, navigate into it by choosing the application name from the table.</p> <p> <b>Note</b> The Cloud Foundry restage action cannot be performed from the cockpit. If you'd like to restage your application, you must use the CF CLI. To learn more about restarting and restaging applications in Cloud Foundry, see <a href="https://docs.cloudfoundry.org/devguide/deploy-apps/start-restart-restage.html">https://docs.cloudfoundry.org/devguide/deploy-apps/start-restart-restage.html</a>.</p>

## Related Information

[Deploy an Application \[page 2491\]](#)

### 6.2.6.3 Add or Remove Application Instances

To increase the availability of your Cloud Foundry application, you can run multiple instances of it.

#### Prerequisites

You must have one application deployed in your Cloud Foundry space.

#### Procedure

1. Navigate to the [Applications](#) page in your Cloud Foundry space.
2. Choose the application for which you want to add or remove application instances.
3. In the [Overview](#) page of your application choose:
  - *Instance* to start an additional instance of your application
  - *Instance* to remove one of the running instances of your application.

#### Related Information

[Deploy an Application \[page 2491\]](#)

### 6.2.6.4 Configuring Application URLs in the Cloud Foundry CLI

By default, all applications running on SAP BTP are accessed on the **default landscape** domain. According to your needs, you can change the default application URL by configuring additional application domains.

The URL for an application deployed on SAP BTP in the Cloud Foundry environment is `https://<application>.cfapps.<region>.hana.ondemand.com`. The domain depends on your location, in the European region, for example, the domain is `cfapps.eu10.hana.ondemand.com`. So, if you're deploying an application with the name "myapp", the default application URL is `https://myapp.cfapps.eu10.hana.ondemand.com`.

Running on the China (Shanghai) region:

There's no default URL available in China, therefore you can't deploy an application without configuring a custom domain first. Please refer to the related information link on how to use custom domains.

#### Note

To check what domains are available in your Cloud Foundry organization, use the command `cf domains`.

## Custom Domains

Use custom domains to reach your applications on your own domain instead of the default domain, for example, `https://myapp.mydomain.com` instead of `https://myapp.cfapps.eu10.hana.ondemand.com`. When you use a custom domain, both the domain name and the TLS/SSL certificate of the server are owned by the customer.

You can configure custom domains using the Cloud Foundry command-line interface with the plugin for custom domains.

For more information, read the [Custom Domain service user guide](#).

## Application Routes

If you want to make your application reachable on another route, you can add additional routes using the Cloud Foundry CLI. If your application is available under the route `https://myapp.cfapps.eu10.hana.ondemand.com`, you could add a route that reads `https://expenses.cfapps.eu10.hana.ondemand.com` that leads to the same application.

### Related Information

[Using Custom Domains \[page 2501\]](#)

[About Routes in the Cockpit \[page 2495\]](#)

[Configuring Application URLs \(Neo environment\)](#)

### 6.2.6.5 About Routes in the Cockpit

To enable your end users to reach your application, create a route and map it to the application in the SAP BTP cockpit.

#### What Is a Route?

A route is the URL that enables your end users to reach your application. Routes belong to a space, and therefore are managed at space level.

The Router component in Cloud Foundry is responsible for routing routes. It maintains a list of mapped applications and compares each request with the list to find the best match. Based on this comparison, it then routes requests to the appropriate application instance.

Currently, you can create, map, and delete HTTP routes. An HTTP route includes a domain, a host name (or subdomain), and an optional path.

## **Creating Routes**

The number of routes you can create in a space depends on your subaccount entitlements and quotas, or on the space quota assigned to that space.

For more information, see [Create Routes \[page 2496\]](#).

## **Mapping Routes to Applications**

You can only map a route to an application that belongs to the same space.

It is possible to map a single route to multiple applications, as well as multiple routes to a single application.

For more information, see [Map Routes to Applications \[page 2498\]](#).

## **Binding Routes to Service Instances**

To bind a route to a service instance in the SAP BTP cockpit, see [Bind Routes to Service Instances \[page 2499\]](#).

### **6.2.6.5.1 Create Routes**

You can configure the URLs through which end users can reach your applications.

#### **Prerequisites**

You must have the Space Developer or the Space Supporter role.

#### **Context**

Routes belong to a space but they're globally unique, regardless of the organization that controls a space. If a route with a URL exists, you can't create a route with the same URL.

#### **Procedure**

1. Navigate to your Cloud Foundry space.

2. Choose *Routes* from the left hand-side navigation.
3. Choose *Create Route*.
4. In the dialog, enter the following parameters:

Parameter	Details
<b>Domain</b>	<p>From the dropdown menu, you can choose either a shared domain (for example, the default <code>cfapps.&lt;region&gt;.hana.ondemand.com</code>) or a private domain that you've previously created using the CF CLI.</p> <p>From the dropdown menu, you can choose either a shared domain or a private domain that you've previously created using the CF CLI.</p> <p>For more information on private domains, see <a href="https://docs.cloudfoundry.org/devguide/deploy-apps/routes-domains.html#private-domains">https://docs.cloudfoundry.org/devguide/deploy-apps/routes-domains.html#private-domains</a>.</p>
<b>Host Name</b>	<p>The host name is your desired subdomain. In the URL, it's added before the selected domain, as follows:  <code>https://&lt;host name&gt;.&lt;domain&gt;</code></p> <p>The host name can contain up to 63 characters.</p>
<b>Path (Optional)</b>	<p>In addition to the domain and subdomain, you can also add a path. You can use paths if you want to create routes for multiple applications available for the same host name and domain. The path becomes part of the URL as follows:</p> <p><code>https://&lt;host name&gt;.&lt;domain&gt;/&lt;path&gt;</code></p>

You can see the preview of your route at the bottom of the dialog.

5. When you're happy with your route, choose *Create*.

## Next Steps

- Once you've created a route, you must map it to your application. For more information, see [Map Routes to Applications \[page 2498\]](#).
- Additionally, you also have the option to bind it to a route service instance by choosing  (*Bind Route Service*) from the *Actions* column. For more information, see [Bind Routes to Service Instances \[page 2499\]](#).

## Related Information

[About Routes in the Cockpit \[page 2495\]](#)

## 6.2.6.5.2 Map Routes to Applications

Once a route has been created, you must map it to an application to make this application reachable for end users.

### Prerequisites

- You have at least one route and one deployed application in the same Cloud Foundry space.
- You must have the Space Developer or the Space Supporter role.

### Procedure

1. Navigate to the [Routes](#) page in your Cloud Foundry space in the cockpit.
2. For the route that you wish to map to an application, choose  ([Map Route](#)) from the [Actions](#) column and select the application you want to map it to.

### Results

Your application can now be accessed via the route mapped to it. You can launch your mapped route from two different places in the cockpit:

- [Routes](#) page:  
Choose  ([Launch Route](#)) from the [Actions](#) column.
- [Overview](#) page of the mapped application:  
Choose the URL from the [Application Routes](#) section.

### Related Information

[About Routes in the Cockpit \[page 2495\]](#)

[Create Routes \[page 2496\]](#)

## 6.2.6.5.3 Bind Routes to Service Instances

You can bind a route to a service instance in the SAP BTP cockpit.

### Prerequisites

In SAP BTP, Cloud Foundry environment, only user-provided and fully-brokered services are currently supported.

- You must have the Space Developer or the Space Supporter role.
- The service instance and the route are in the same Cloud Foundry space.
- To bind a route to a user-provided service instance, the service instance must include the URL of the route service. To include it, set the `route_service_url` property.
- To bind a route to a managed service instance, make sure this service can be bound by setting route forwarding in the `requires` property:  
`requires=[route_forwarding]`

### Context

Before you proceed with the route binding, keep in mind the following:

- You can have a route bound to an application and a service instance at the same time.
- You can bind one or multiple routes to a single service instance.

Now that you're aware of these specifics, let's continue with the route binding steps.

### Procedure

1. Navigate to the [Routes](#) page in your Cloud Foundry space in the cockpit.
2. Choose  ([Bind Route Service](#)) from the [Actions](#) column.
3. Select a service instance from the dropdown.

The dropdown contains a list of all service instances that you can bind to the route. If an existing service instance can't be bound to a route, you won't see it in the dropdown.

If you want to create a new service instance from your services available in the Service Marketplace, see [Creating Service Instances in Cloud Foundry](#).

To create a new user-provided service instance, see [Creating User-Provided Service Instances in Cloud Foundry Environment](#).

4. (Optional) If you want to customize the route binding service, mark the [Specify Parameters](#) checkbox.

You have the option to either upload your own JSON file or you can directly enter JSON parameters and their values in the field below.

For more information about what parameters you can use, see <https://v3-apidocs.cloudfoundry.org/version/3.138.0/index.html#service-route-binding>.

5. Choose *Bind*.

## Related Information

[About Routes in the Cockpit \[page 2495\]](#)

### 6.2.6.6 Security Groups

In the SAP BTP, Cloud Foundry environment, application security groups represent a collection of rules that specify the protocols, ports, and IP address ranges where an application or a task instance sends traffic. You can access this information in the SAP BTP cockpit.

#### Overview

In the SAP BTP, Cloud Foundry environment, the security groups are space-scoped and dynamic. They are also split into:

- Default security groups:
  - dns
  - public\_networks
- Service-related security groups  
A service-related security group is created when you bind the respective service to an application. Each service-related security group is specific to a single SAP BTP service. The name of the service is part of the security group name. For example, <service-name>-security-group.

For more information, see <https://docs.cloudfoundry.org/concepts/asg.html>.

#### How to Find the Security Groups Page?

1. In the SAP BTP cockpit, navigate to your subaccount.  
For more information, see [Navigate in the Cockpit](#).
2. In the navigation menu, choose  *Cloud Foundry*  *Spaces*.
3. Go to one of your spaces.
4. In the navigation menu, choose *Security Groups*.

## What Can You Do on the Security Groups Page?

The security groups manage the outgoing traffic of your application by allowing access to the network based on a list of specific protocols, ports, and IP address ranges. You have read-only access to them.

If your application is trying to access an endpoint not specified by the security group, the access is denied.

On the *Security Groups* page in the cockpit, you can check the information in the *Rules* table to verify that the application is using the correct protocols, ports, and IP address ranges.

### ⓘ Note

Instead of using the cockpit, you can get information about a security group by running the corresponding CLI command. For more information, see <https://cli.cloudfoundry.org/en-US/v8/>.

### 6.2.6.7 Using Custom Domains

SAP BTP allows subaccount owners to make their SAP BTP applications reachable and secure via a custom domain that is different from the default domain – for example, **subdomain.mydomain.com**.

See the documentation for the [Custom Domain service](#) for more information about configuring and managing custom domains.

## 6.2.7 Audit Logging in the Cloud Foundry Environment

In this section, you can find information for audit log functionalities in the Cloud Foundry environment.

SAP Audit Log is a core, security, and compliance-based SAP BTP service to provide means for audit purposes. The following features of Audit Log Service are available for SAP BTP Applications and Services:

- With the default features, you have:
  - Written compliance audit data from SAP BTP services and applications, through OAuth2 service plan.
  - Audit data securely stored for a default retention time of 90 days with no additional costs applied.
  - Retrieval of audit data within the default retention period ensured as part of the Auditlog Management Service.
- With the advanced features you can:
  - Write audit data from your owned BTP applications.
  - Configure the retention period.

The advanced features enable you to comply with the SAP Product Standards and Business Industry regulations you're subject to. To enable the advanced features, you need to enable the premium edition service plan, where additional costs are applied based on the consumed volumes.

Audit logs are a special type of logs. They represent security-relevant chronological records that provide documentary evidence for an event or activity. The following table gives you a better understanding what is the difference between audit logs, activity logs, and application logs. The SAP Audit Log only stores audit logs written by SAP BTP services, when you take action over your account data.

	 Audit logs	 Activity logs	 Application logs
Scope	Evidence	History	Fix Issues
Used by	 Auditor	 User	 Developer
Tamper proof	✓		
UI performance		✓	
Frequent data access		✓	
Long data retention	✓		
Secure low cost architecture	✓		

The SAP Audit Log stores audit logs representing different actions taken over your account and/or data. There are predefined audit categories, which represent such kinds of actions:

- Data protection and privacy related
  - *audit.data-access* read-access logging records for access to sensitive personal data;
  - *audit.data-modification* data modification logging records for sensitive personal data.
- Security related
  - *audit.security-events* logging of general security events like login, logout, and other;
  - *audit.configuration* logging of security critical configuration changes.

See [Change Logging and Read-Access Logging \[page 3113\]](#).

If your subaccount audit logs contain other log types or information, create a [support ticket](#) with component BC-CP-CF-SEC-AUDITLG.

## Related Information

[Audit Log Retrieval API Usage for Subaccounts in the Cloud Foundry Environment \[page 2503\]](#)

[Audit Log Retention for the Cloud Foundry Environment \[page 2515\]](#)

[SAP Audit Log Viewer service for the Cloud Foundry Environment \[page 2528\]](#)

[Guided Answers for Audit Logging in the Cloud Foundry Environment](#)

## 6.2.7.1 Audit Log Retrieval API Usage for Subaccounts in the Cloud Foundry Environment

The audit log retrieval API allows you to retrieve the audit logs for your SAP BTP Cloud Foundry environment subaccount.

### ⓘ Note

After the audit logs are generated and sent to the Audit Log service, the logs are not immediately accessible for retrieval.

Retrieving of audit logs via the Audit Log Retrieval API is limited to the size of the audit logs generated for the subaccount. The audit log results are provided as a collection of JSON entities. The possible request rate is now limited to:

- 4 requests per second per auth-token/tenant on CF-EU11, CF-EU30, CF-US20, CF-US21, CF-JP10, CF-JP20, CF-BR10. There is also a burst queue that can temporarily allow up to 20 requests per second.
- 8 requests per second per auth-token/tenant on CF-EU10, CF-EU20, CF-US10, CF-US30. There is also a burst queue that can temporarily allow up to 40 requests per second.

The API returns a HTTP 429 response code when the limits are exceeded.

The audit log retrieval API is protected by OAuth. To consume the audit log retrieval API, you need your system to generate and use a valid OAuth.

## Prerequisites

To execute the procedure, you need to have the `Space Developer` role for the corresponding Space.

### ⓘ Note

For security reasons, we recommend you to re-create the `auditlog-management` Service-Instance bindings and service keys at least every 90 days.

## Create Instance of the `auditlog-management` Service

1. Create a Cloud Foundry Org and Space, in case you don't have any. See [Create Spaces \[page 2441\]](#).
2. Log in the Cloud Foundry landscape using the corresponding Cloud Foundry API (Infrastructure/Landscape Overview).

```
cf login -a <API_URL> -o <ORG> -s <SPACE> -u <USER>
```

3. Create a service instance of the service `auditlog-management`.

### ⓘ Note

For security reasons, we strongly recommend you to adopt the provided mutual TLS authentication (mTLS). The mTLS authentication relies on X.509 certificates for the verification of the parties in

the network connection. Create the auditlog-management service instances with the additional parameters, as explained in the following mTLS points.

- (Recommended) For mTLS authentication using X.509 certificates, use:

```
cf create-service auditlog-management default <SERVICE_INSTANCE> -c '{  
  "xs-security": {  
    "xsappname": "auditlog",  
    "oauth2-configuration": {  
      "credential-types": [  
        "x509"  
      ],  
      "grant-types": [  
        "client_credentials"  
      ]  
    }  
  }'  
}'
```

#### ⓘ Note

The `xsappname` attribute from the service parameters JSON needs to have unique instance for each service. The `xsappname` attribute can hold an arbitrary string as value, which is not required to be `auditlog`. The code block serves only as example.

- For non-mTLS authentication, use

```
cf create-service auditlog-management default <SERVICE_INSTANCE>
```

#### ⓘ Note

If there's a previously created auditlog-management service instance without the security parameters for mTLS, we recommend you to delete that instance and create a new instance with mTLS authentication using the parameters from the previous (recommended) scenario.

#### 4. Create a key for the service instance:

- For service instances with mTLS, use:

```
cf create-service-key <SERVICE_INSTANCE> <SERVICE_KEY> -c '{  
  "xsuaa": {  
    "credential-type": "x509",  
    "x509": {  
      "key-length": 2048,  
      "validity": 1,  
      "validity-type": "DAYS"  
    }  
  }'  
}'
```

#### ⓘ Note

You can specify the validity period of the service key for each use case. To use the service instance, once the key validity expires, create a new key. By default and for security reasons, the key from this code example is created with a validity of 1 day.

- For service instances without mTLS, use:

```
cf create-service-key <SERVICE_INSTANCE> <SERVICE_KEY>
```

#### 5. List the key of the service instance:

```
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY>
```

6. Extract data from the service key.

- For mTLS scenario:
  - Extract the values for `uaa.clientid` and `uaa.certurl` of the key of the service instance for access token creation.
  - Extract the value for `url` and use it for later request to retrieve audit logs.
  - Extract the value for `uaa.certificate`, remove all `\n` entries from the X.509 certificate, and save it as a file in the `.pem` format.
  - Extract the value for `uaa.key`, remove all `\n` entries from the RSA private key, and save it as a file in the `.pem` format.

 Note

To extract of the values and avoid errors while copying or removing characters, you can use the `sed` and `jq` tools:

```
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.certurl  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.clientid  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .url  
# the following two commands save the output to .pem files on the  
location, where the command is executed  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.certificate > mtls-certificate.pem  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.key > mtls-private-key.pem
```

- For non-mTLS scenario:

Extract values for `uaa.url`, `uaa.clientid` and `uaa.clientsecret` of the key of the service instance for access token creation. Extract the `url` to be used for later request to retrieve audit logs.

## Create an OAuth Access Token

1. Request an OAuth access token

- For mTLS scenario

To request the OAuth access token via the mTLS extracted X.509 certificates and RSA private key flow, execute an HTTP POST request with the following parameters:

```
URL: <value of "uaa.certurl">/oauth/token?  
grant_type=client_credentials&client_id=<value of "uaa.clientid">  
Method: POST  
Authentication Method: Client Certificates  
Certificate: <Value of "uaa.certificate">  
Key: <Value of "uaa.key">  
grant_type: client_credentials  
client_id: <Value of "uaa.clientid">
```

Example:

```
curl --cert <path to certificate.pem> --key <path to key.pem>  
--request POST https://<value of "uaa.certurl">/oauth/token -d  
'grant_type=client_credentials&client_id=<Value of "uaa.clientid">'
```

- For non-mTLS scenario

Request the OAuth access token via the OAuth client credentials flow, by executing an HTTP POST request with the following parameters:

```
URL: <value of "uaa.url">/oauth/token?grant_type=client_credentials
Method: POST
Authentication Method: Basic Authentication
Username: <Value of "uaa.clientid">
Password: <Value of "uaa.clientsecret">
```

2. Extract the value of the `access_token` attribute from the JSON response.

This token grants access to the audit logs of the subaccount on the landscape where the service instance is created.

#### ⓘ Note

The token is valid for 1 hour.

#### ⓘ Note

This token grants access to the audit logs of the subaccount on the landscape. The content of the access token can be displayed using a standard JWT decoder

## Audit Log Retrieval

The Audit Log Retrieval API incorporates server-side paging to address the efficient handling of queries producing substantial result sets.

#### ⓘ Note

The logs are not immediately accessible for retrieval

In instances where the query produces a result with significant size, the API systematically partitions the output into manageable chunks. The response includes an HTTP header featuring a handle, designed to facilitate the retrieval of subsequent result segments. To obtain additional portions of the result, pass the handle as a URL parameter in the subsequent retrieval request. The size of each chunk is predetermined, with a page size fixed at 500.

Supported request parameters:

- `time_from` and `time_to` – if no time filter is specified the default timeframe of 30 days back is returned. Use the following format: YYYY-MM-DDTHH:MM:SS, for example 2018-05-11T10:42:00. The time is in UTC.
- `handle` - in case the result set is too large, it's chunked, and a handle is returned for reading the next chunk. The handle is returned as a Response Header in the form: `Paging: handle=<value>`. Then you can provide `handle=<value>` as a request parameter in the subsequent retrieval request.
- `category` - this parameter allows you to filter the audit logs by predefined event types. If no event type is specified, all event types are returned. The available event types are:
  - `audit.security-events`
  - `audit.configuration`

- **audit.data-access**
- **audit.data-modification**

To filter by a specific event type, provide the event type name as the value for this parameter, for example `category=audit.data-access`. Multiple event types can be specified separated by commas, for example `category=audit.data-access,audit.configuration`.

#### Note

Filtering by category doesn't work for custom retention periods and Sovereign cloud. In such case, an error code 501 is returned with the following format and message:

```
{
  "error": {
    "code": "filter_category_not_supported",
    "message": "Filtering by category not available for custom retention periods. Remove categories to proceed."
  }
}
```

### **Example: Get audit logs for the last 30 days**

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords
```

Provide the OAuth access token as an "Authorization" header: "Authorization: Bearer <access\_token>"

The response is in JSON format, containing the audit log entries, split on pages if needed.

### **Example: Get audit logs filtering by time**

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords?
time_from=2018-05-10T10:42:00&time_to=2018-05-11T10:46:00
```

### **Example: Get audit logs filtered by event type**

You can query for one of the predefined audit log event types for the `category` parameter.

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords?
time_from=2018-05-10T10:42:00&time_to=2018-05-11T10:46:00&category=audit.data-access
```

### **Example: Get audit logs next chunk determined by the server-side paging**

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords?
handle=2018-06-14T10:11:18.968<4f932695-8616-4e1f-ac9a-1cdfb758f01d<2018-06-14T10:11:00.000
```

Response codes:

HTTP Codes

HTTP 200 OK  
HTTP 204 NO\_CONTENT  
HTTP 401 UNAUTHORIZED  
HTTP 400 BAD\_REQUEST  
HTTP 403 FORBIDDEN  
HTTP 429 Too Many Requests  
HTTP 501 NOT\_IMPLEMENTED

## Results

```
{  
  "message_uuid": "e3a533c1-57b3-42b5-b514-8934ec5b6a6a",  
  "time": "2018-10-04T08:12:00.239Z",  
  "tenant": "c6da83a8-dc72-4374-b8f7-42b37a99XXXX",  
  "org_id": "82f1da92-e5b3-4cc5-8c90-964165afXXXX",  
  "space_id": "82f1da92-e5b3-4cc5-8c90-964165aXXXX",  
  "app_or_service_id": "5fb40c3a-d36a-42c7-adc8-e4abd83dXXXX",  
  "als_service_id": "c18f9b6d-a8af-431c-a187-749ebc59XXXX",  
  "user": "test",  
  "category": "audit.security-events",  
  "format_version": "",  
  "message": {  
    "uuid": "e3a533c1-57b3-42b5-b514-8934ec5b6a6a",  
    "user": "test",  
    "time": "2018-10-04T08:12:00.239Z",  
    "ip": "10.58.183.15",  
  
    "data": "{\"level\":\"INFO\", \"origin\":null, \"msgNo\":1, \"msgId\":\"a2cf08ee-eedd-455b-bbd6-400d6b611116\", \"message\": \"ClientAuthenticationSuccess ('Client authentication success'): principal=sb-40ae21f3-5034-40dd-ac0d-0c9d3e0ebb06!b3034|auditlog-management!b3034, origin=[remoteAddress=52.58.183.15, clientId=sb-40ae21f3-5034-40dd-ac0d-0c9d3e0ebb06!b3034|auditlog-management!b3034], identityZoneId=[c6da83a8-dc72-4374-b8f7-42b37a99db2b]\", \"user\":null, \"version\":\"1.0\"}",  
    "id": "cb046a0f-cc23-406b-a1d2-22ee6cf89a4d",  
    "category": "audit.security-events",  
    "tenant": "c6da83a8-dc72-4374-b8f7-42b37a99XXXX"  
  }  
}
```

## Related Information

[Audit Log Retrieval API for the Cloud Foundry Environment](#)  
[API Documentation](#)

## 6.2.7.2 Audit Log Retrieval API for Global Accounts in the Cloud Foundry Environment

On Central regions, Audit Log Retrieval API allows you to retrieve audit logs written on behalf of your SAP BTP Global Account.

Your central region is the eu10 Europe (Frankfurt) region, unless your global account is located in the China (Shanghai) and Government Cloud (US) regions. Then use the landscape domain of your central region.

Audit Log Retrieval API is protected by OAuth, and to consume it, you need your system to generate and use a valid OAuth.

### ⓘ Note

After the audit logs are generated and sent to the Audit Log service, the logs are not immediately accessible for retrieval.

Retrieving of audit logs via Audit Log Retrieval API is limited to the size of the audit logs written on behalf of the global account. The possible request rate is now limited to:

- 4 requests per second per auth-token/tenant on CF-EU11, CF-EU30, CF-US20, CF-US21, CF-JP10, CF-JP20, CF-BR10. There is also a burst queue that can temporarily allow up to 20 requests per second.
- 8 requests per second per auth-token/tenant on CF-EU10, CF-EU20, CF-US10, CF-US30. There is also a burst queue that can temporarily allow up to 40 requests per second.

The API returns a HTTP 429 response code when the limit is exceeded.

### ⓘ Note

Custom managed key (CMK) is not supported for audit logs generated on Global Account level.

## Prerequisites

You do the following procedure through Cloud Foundry Environment, which is part of a subaccount. To enable Audit Log Retrieval API for Global Account , you must perform the following prerequisite steps as a global account administrator:

1. You need to have one or more subaccounts in the Central region<sup>(1)</sup>. Create one if needed.

### ⓘ Note

ALL space developer users can access the service instance in the subaccount where it's entitled. To restrict access to your global account audit logs we recommend, you create a new subaccount in the Central regions<sup>(1)</sup> and add only users eligible to read global account audit logs to it.

2. Follow the steps to entitle one or more of your subaccounts with the **Central plan** of service Auditlog Management:
  1. Navigate to your global account.
  2. Choose  [Entitlements](#)  [Entity Assignments](#) from the left-hand side menu.
  3. In the pop-up window, select all the subaccounts for which you would like to configure or display entitlements.

4. Choose *Select* to apply the filter.  
You get a table for each of the subaccounts you selected, which displays the current entitlement and quota assignments.
5. Choose *Configure Entitlements* to start editing entitlements for a particular subaccount. You can now edit the entitlements table.

**ⓘ Note**

You can edit entitlements for only one subaccount at a time.

6. Choose *Add Service Plans*.
  7. Select *Auditlog Management* from the list on the left-hand side.
  8. Choose the checkbox of *Central plan* on the right-hand side.
  9. Choose *Add 1 Service Plan*.
  10. Choose *Save* and exit edit mode for that subaccount.
3. To execute the procedure, you need to have the `Space Developer` role for the corresponding Space.

**ⓘ Note**

For security reasons, we strongly recommend you to re-create the auditlog-management Service-Instance bindings and service keys at least every 90 days.

## Create Instance of the auditlog-management Service

1. Create a Cloud Foundry Org and Space, in case you don't have any. See [Create Spaces \[page 2441\]](#).
2. Log in the Cloud Foundry landscape using the corresponding Cloud Foundry API (Infrastructure/Landscape Overview).

```
cf login -a <API_URL> -o <ORG> -s <SPACE> -u <USER>
```

3. Create a service instance of the service auditlog-management:

**ⓘ Note**

For security reasons, we strongly recommend you to adopt the provided mutual TLS authentication (mTLS). The mTLS authentication relies on X.509 certificates for the verification of the parties in the network connection. Create the auditlog-management service instances with the additional parameters, as explained in the following mTLS points.

- (Recommended) For mTLS authentication using X.509 certificates, use:

```
cf create-service auditlog-management central <SERVICE_INSTANCE> -c '{
  "xs-security": {
    "xsappname": "auditlog",
    "oauth2-configuration": {
      "credential-types": [
        "x509"
      ],
      "grant-types": [
        "client_credentials"
      ]
    }
  }
}'
```

```
}
```

### ⓘ Note

The `xsappname` attribute from the service parameters JSON needs to have unique instance for each service. The `xsappname` attribute can hold an arbitrary string as value, which is not required to be `auditlog`. The code block serves only as example.

- For non-mTLS authentication, use

```
cf create-service auditlog-management central <SERVICE_INSTANCE>
```

### ⓘ Note

If there's a previously created auditlog-management service instance without the security parameters for mTLS, we recommend you to delete that instance and create a new instance with mTLS authentication using the parameters from the previous (recommended) scenario.

#### 4. Create a key for the service instance:

- For service instances with mTLS, use:

```
cf create-service-key <SERVICE_INSTANCE> <SERVICE_KEY> -c '{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "key-length": 2048,
      "validity": 1,
      "validity-type": "DAYS"
    }
  }
}'
```

### ⓘ Note

You can specify the validity period of the service key for each use case. To use the service instance, once the key validity expires, create a new key. By default and for security reasons, the key from this code example is created with a validity of 1 day.

- For service instances without mTLS, use:

```
cf create-service-key <SERVICE_INSTANCE> <SERVICE_KEY>
```

#### 5. List the key of the service instance:

```
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY>
```

#### 6. Extract data from the service key.

- For mTLS scenario:

- Extract the values for `uaa.clientid` and `uaa.certurl` of the key of the service instance for access token creation.
- Extract the value for `url` and use it for later request to retrieve audit logs.
- Extract the value for `uaa.certificate`, remove all `\n` entries from the X.509 certificate, and save it as a file in the `.pem` format.
- Extract the value for `uaa.key`, remove all `\n` entries from the RSA private key, and save it as a file in the `.pem` format.

### Note

To extract of the values and avoid errors while copying or removing characters, you can use the sed and jq tools:

```
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.certurl  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.clientid  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .url  
# the following two commands save the output to .pem files on the  
location, where the command is executed  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.certificate > mtls-certificate.pem  
cf service-key <SERVICE_INSTANCE> <SERVICE_KEY> | sed '/Getting key/d'  
| jq --raw-output .uaa.key > mtls-private-key.pem
```

- For non-mTLS scenario:

Extract values for uaa.url, uaa.clientid and uaa.clientsecret of the key of the service instance for access token creation. Extract the url to be used for later request to retrieve audit logs.

## Create an OAuth Access Token

### 1. Request an OAuth access token

- For mTLS scenario

To request the OAuth access token via the mTLS extracted X.509 certificates and RSA private key flow, execute an HTTP POST request with the following parameters:

```
URL: <value of "uaa.certurl">/oauth/token?  
grant_type=client_credentials&client_id=<value of "uaa.clientid">  
Method: POST  
Authentication Method: Client Certificates  
Certificate: <Value of "uaa.certificate">  
Key: <Value of "uaa.key">  
grant_type: client_credentials  
client_id: <Value of "uaa.clientid">
```

Example:

```
curl --cert <path to certificate.pem> --key <path to key.pem>  
--request POST https://<value of "uaa.certurl">/oauth/token -d  
'grant_type=client_credentials&client_id=<Value of "uaa.clientid">'
```

- For non-mTLS scenario

Request the OAuth access token via the OAuth client credentials flow, by executing an HTTP POST request with the following parameters:

```
URL: <value of "uaa.url">/oauth/token?grant_type=client_credentials  
Method: POST  
Authentication Method: Basic Authentication  
Username: <Value of "uaa.clientid">  
Password: <Value of "uaa.clientsecret">
```

### 2. Extract the value of the access\_token attribute from the JSON response.

This token grants access to the audit logs of the subaccount on the landscape where the service instance is created.

**ⓘ Note**

The token is valid for 12 hours.

**ⓘ Note**

The content of the access token can be displayed using a standard JWT decoder

## Audit Log Retrieval

The Audit Log Retrieval API incorporates server-side paging to address the efficient handling of queries producing substantial result sets.

**ⓘ Note**

The logs are not immediately accessible for retrieval

In instances where the query produces a result with significant size, the API systematically partitions the output into manageable chunks. The response includes an HTTP header featuring a handle, designed to facilitate the retrieval of subsequent result segments. To obtain additional portions of the result, pass the handle as a URL parameter in the subsequent retrieval request. The size of each chunk is predetermined, with a page size fixed at 500.

Supported request parameters:

- `time_from` and `time_to` – if no time filter is specified the default timeframe of 30 days back is returned. Use the following format: 2018-05-11T10:42:00. The time is in UTC.
- `handle` – in case the result set is too large, it's chunked, and a handle is returned for reading the next chunk. The handle is returned as a Response Header in the form: `Paging: handle=<value>`. Then you can provide `handle=<value>` as a request parameter in the subsequent retrieval request.

**ⓘ Note**

The `handle` parameter is a base 64 encoded string. Don't attempt to decode it. Send the `handle` in the subsequent request exactly as received in the previous one, without any modification.

### Example: Get audit logs for the last 30 days

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords
```

Provide the OAuth access token as an "Authorization" header: "Authorization: Bearer <access\_token>"

The response is in JSON format, containing the audit log entries, split on pages if needed.

### Example: Get audit logs filtering by time

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords?  
time_from=2018-05-10T10:42:00&time_to=2018-05-11T10:46:00
```

## Example: Get audit logs next chunk determined by the server-side paging

Execute the following HTTP GET request:

```
<url_from_service_key>/auditlog/v2/auditlogrecords?  
time_from=2018-05-10T10:42:00&time_to=2018-05-11T10:46:00&handle= <handle value>
```

Response codes:

HTTP Codes

HTTP 200 OK

HTTP 204 NO\_CONTENT

HTTP 401 UNAUTHORIZED

HTTP 400 BAD\_REQUEST

HTTP 403 FORBIDDEN

HTTP 429 Too Many Requests

Predefined audit log message categories:

```
'audit.security-events'  
'audit.configuration'  
'audit.data-access'  
'audit.data-modification'
```

## Results

```
{  
  "message_uuid": "e3a533c1-57b3-42b5-b514-8934ec5b6a6a",  
  "time": "2018-10-04T08:12:00.239Z",  
  "tenant": "c6da83a8-dc72-4374-b8f7-42b37a99XXXX",  
  "org_id": "82f1da92-e5b3-4cc5-8c90-964165afXXXX",  
  "space_id": "82f1da92-e5b3-4cc5-8c90-964165aXXXX",  
  "app_or_service_id": "5fb40c3a-d36a-42c7-adc8-e4abd83dXXXX",  
  "als_service_id": "c18f9b6d-a8af-431c-a187-749ebc59XXXX",  
  "user": "test",  
  "category": "audit.security-events",  
  "format_version": "",  
  "message": {  
    "uuid": " e3a533c1-57b3-42b5-b514-8934ec5b6a6a ",  
    "user": "test",  
    "time": "2018-10-04T08:12:00.239Z",  
    "ip": "10.58.183.15",  
  
    "data": "{\"level\":\"INFO\", \"origin\":null, \"msgNo\":1, \"msgId\":\"a2cf08ee-  
eedd-455b-bbd6- 400d6b611116\", \"message\":\"ClientAuthenticationSuccess  
('Client authentication success'): principal=sb-40ae21f3-5034-40dd-  
ac0d-0c9d3e0ebb06!b3034|auditlog-management!b3034,  
origin=[remoteAddress=52.58.183.15, clientId=sb-40ae21f3-5034-40dd-  
ac0d-0c9d3e0ebb06!b3034|auditlog-management!b3034], identityZoneId=[c6da83a8-  
dc72-4374-b8f7-42b37a99db2b]\", \"user\":null, \"version\":\"1.0\"}",  
    "id": "cb046a0f-cc23-406b-a1d2-22ee6cf89a4d",  
    "category": "audit.security-events",  
    "tenant": "c6da83a8-dc72-4374-b8f7-42b37a99XXXX"
```

```
    }
```

## 6.2.7.3 Audit Log Retention for the Cloud Foundry Environment

The audit log data is stored on a subaccount level. The access to the stored audit log data is strictly restricted - only authorized stakeholders can preview, retrieve, and download their audit log data.

Currently, there are two main categories of retention of audit data, divided by the source of generation:

- [Retention of Audit Data written from SAP BTP Applications and Services \[page 2515\]](#) (SAP-generated)
- [Retention of Audit Data written from Customer's BTP Applications \[page 2516\]](#) (Customer-generated)

### 6.2.7.3.1 Retention of Audit Data written from SAP BTP Applications and Services

#### Free Retention Period

By default, you can retain the securely stored SAP-generated audit log data for your subaccount for 90 days, free of charge. After this period, the data is automatically deleted. Within this free default retention period, as an authorized stakeholder you can access, retrieve, and download the stored audit log data.

#### Configurable Retention Period

##### **Enable *Audit Log service, premium edition***

To follow the Industry and Business Regulations requirements for a longer retention and storage of audit log data written on behalf of a customer's subaccount, and to be able to configure a custom retention period longer than the free 90 days, you must enable [\*Audit Log service, premium edition\*](#) service plan for your subaccount as follows:

1. Entitle your subaccount with service auditlog and plan premium. For more information, see [Configure Entitlements and Quotas for Subaccounts \[page 2193\]](#)
2. Create a service instance *auditlog* of plan *premium*. For more information, see [Creating Service Instances \[page 250\]](#)

##### **Configure a Retention period**

When you enable the [\*Audit Log service, premium edition\*](#) service plan, you can create a [support ticket](#) with component BC-CP-CF-SEC-AUDITLG to the Audit Log service to configure a flexible retention period for storing your audit data. In the support ticket provide the following data:

- Subaccount ID(s) and Tenant ID(s) whose retention you want to extend.
- Your desired retention period in days.
- The date from which you want the configurable retention to be effective.

## Pricing

The premium edition with its advanced functionalities is a commercialized consumption based service plan, which prices per GB you can check in [SAP Discovery Center](#).

## Revoking Custom Retention of SAP-Generated Audit Data

### Revoke Custom retention of an Existing Subaccount

To revoke custom retention, create a [support ticket](#) with component BC-CP-CF-SEC-AUDITLG to the Audit Log service to configure a flexible retention period for storing your audit data. In the support ticket provide the following data:

- Subaccount ID(s) and Tenant ID(s) whose custom retention you want to revoke.
- The date from which you want the revocation to be effective.

When you revoke custom retention all stored audit data which is older than 90 days is removed and charging for extra storage stops.

### Revoke Custom retention of a Terminated Subaccount

SAP-generated audit log data for a terminated subaccount is deleted when it is older than 90 days. The 90-day retention period is kept free of charge for audit purposes.

#### ⓘ Note

When you delete a subaccount, all audit log data older than 90 days is automatically deleted.

#### ⓘ Note

When you delete an auditlog premium service instance, the charging for storage doesn't stop automatically. To avoid accumulating further costs, you must either create a [support ticket](#) to revoke the custom retention period, or delete the subaccount.

## 6.2.7.3.2 Retention of Audit Data written from Customer's BTP Applications

### Initial retention period

There is no free retention period for any customer-generated audit data. If you enable **Audit Log service, premium edition** service plan, the default period in which you can retain all audit data written from your

customer applications built on top of SAP BTP is 90 days. For more information, see [Audit Log Write API for Customers \[page 2518\]](#).

## Configurable Retention Period for Subaccounts

When you have enabled the **Audit Log service, premium edition** service plan, you can create a support ticket with component BC-CP-CF-SEC-AUDITLG to the Audit Log service to configure a custom retention period for storing your audit data.

## Availability of the Write API for Customers

The functionality for writing customer-generated audit data as part of the **Audit Log service, premium edition** service plan, is currently available in the following regions:

- Europe (Frankfurt)
- Europe (Frankfurt, EU Access)
- US East (VA)

If you want to use this functionality in another region, contact the Service owner to discuss the conditions for storing large volumes of audit data. **Audit Log service, premium edition** with its advanced functionalities is a commercialized consumption-based service plan. You can find the cost per Customer Written Volume stored in GB in [SAP Discovery Center](#).

## Revoking Custom Retention of SAP-Generated Audit Data

### Revoke Custom retention of an Existing Subaccount

To revoke custom retention, create a [support ticket](#) with component BC-CP-CF-SEC-AUDITLG to the Audit Log service to configure a flexible retention period for storing your audit data. In the support ticket provide the following data:

- Subaccount ID(s) and Tenant ID(s) whose custom retention you want to revoke.
- The date from which you want the revocation to be effective.

When you revoke custom retention all stored audit data which is older than 90 days is removed and charging for extra storage stops.

#### Note

Audit log data written from Customer's SAP BTP Applications is still charged per GB at the price in [SAP Discovery Center](#) no matter the retention period in days.

### Revoke Custom retention of a Terminated Subaccount

Audit log data written by Customer's SAP BTP Applications is removed immediately in case of a subaccount termination, as noted in [Delete a Subaccount \[page 2184\]](#).

#### ⓘ Note

When you delete a subaccount, all audit log data written from your Customer's SAP BTP Applications is automatically deleted.

#### ⓘ Note

When you delete an auditlog premium service instance, the charging for storage doesn't stop automatically. To avoid accumulating further costs, you must either create a [support ticket](#) to revoke the custom retention period, or delete the subaccount.

## 6.2.7.4 Audit Log Write API for Customers

The SAP Audit Log provides an Audit Log Write API for customers, which enables the recording of audit relevant events.

### Prerequisites for Using the Audit Log Write API for Customers

#### Do the Enablement and Entitlements

As a first step to use the Audit Log Write API, make sure that the service is enabled for your subaccount and that there is an entitlement for **Audit Log service, premium edition** service plan.

If the *Audit Log Service* of plan *premium* is not visible from the account cockpit service marketplace for your subaccount, do the following to entitle the service:

- To add entitlements on subaccount level, navigate to the subaccount via the account cockpit and choose [Entitlements](#) [Configure Entitlements](#)
- To add entitlements on global account level via the account cockpit, navigate to the account cockpit global account view and choose [Entitlements](#) [Service Assignments](#)
- To add entitlements for a global account via the SAP BTP Control Center, choose the global account and go to [Edit](#) [Assign Services](#). To select the Auditlog Service checkbox choose [Next](#) [Set Entitlements](#) [Select Auditlog Service](#) and ensure that there is quota for the "premium" plan of the service in the desired region(s).

#### Create a Service Instance

The *premium* plan of the auditlog service uses mTLS authentication via X.509 certificates. To create a service instance via the account cockpit, do the following:

1. Select the *premium* plan for the service instance creation from the Service Marketplace inside the SAP BTP Cockpit.
2. Choose a runtime and a name for the service instance.
3. Specify a unique *xsappname* per subaccount for each instance and provide the x509 authentication parameters during service instance creation.

You can also create a service instance via the CF client with the following sequence of commands:

```
cf api <cf-domain>
cf login
cf target -o <desired-org> -s <desired-space>
cf create-service auditlog premium <my-service-instance> -c '{
  "xs-security": {
    "xsappname": "<some-unique-name>",
    "oauth2-configuration": {
      "credential-types": ["x509"],
      "grant-types": ["client_credentials"]
    }
  }
}'
```

### ⓘ Note

Deleting the service instance doesn't delete the customer audit data already stored in the Audit Log Viewer. For more information, see [Audit Log Retention for the Cloud Foundry Environment \[page 2515\]](#)

## Create a Service Key

The service keys provide credentials and URLs for interacting with your service instance. You can create a service key in one of the following ways: from the account cockpit or via CF command.

- From your account cockpit, select the created *premium* service instance from the *Instances and Subscriptions* section. Depending on the chosen runtime environment, you can see a *Create* button for either a service key or a service binding. In the service key/binding creation box, specify the validity of the X.509 certificates used for authentication.
- In the CF client, use the following command. The example parameters sets the certificate credentials validity to 30 days. After that period, the credentials expire and need to be rotated. For more information about the rotation, see the *Credentials rotation* section.

```
cf create-service-key <my-service> <my-key> -c '{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "key-length": 2048,
      "validity": 30,
      "validity-type": "DAYS"
    }
  }
}'
```

## Create a Service Binding

Like service keys, the classical service bindings provide credentials and endpoints for using a service instance and are mainly intended to be used with Cloud Foundry applications. You can bind a CF application to one or more service instances. As a result, in the VCAP\_SERVICES application environment variable there is a dedicated JSON object for each service binding.

You can perform the binding operation directly from the account cockpit settings of the application, via the CF client, or via the application deployment manifest. Specifically for the Audit Log Write API “premium” plan, it is important to provide the credentials parameter to enable the provisioning of X.509 certificates. Examples:

- Create a service binding via the CF client:

```
cf bind-service <sample-application> <auditlog-premium-service-instance> -c '{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "key-length": 2048,
      "validity": 30,
      "validity-type": "DAYS"
    }
  }
}'
```

```

        "x509": {
            "key-length": 2048,
            "validity": 30,
            "validity-type": "DAYS"
        }
    }
}

```

- Create a service binding through a simple deployment manifest:

```

---
applications:
- name: java-auditlog-sample
  memory: 512M
  instances: 1
  path: target/java-auditlog-sample.war
  env:
    TARGET_RUNTIME: tomcat
  services:
    - name: "<auditlog-premium-service-instance-name>"
      parameters:
        xsuaa:
          credential-type: "x509"
        x509:
          key-length: 2048
          validity: 30
          validity-type: "DAYS"

```

- Create a service binding through an MTA deployment manifest:

```

modules:
- name: <module using the Audit Log Service>
  requires:
    - name: <Audit Log Service Premium Instance Name>
      parameters:
        config:
          xsuaa:
            credential-type: x509
          x509:
            key-length: 2048
            validity: 2
            validity-type: MONTHS
    - name: <other modules>
resources:
- name: <Audit Log Service Premium Instance Name>
  type: org.cloudfoundry.managed-service
  parameters:
    service: auditlog
    service-plan: premium

```

## Obtain Credentials for Authentication and Authorization

This section describes the approach to obtain credentials for authentication and authorization towards the Audit Log Write API.

- Retrieving service key information

When you create a service key or binding, do one of the following:

- Download the service key data from the account cockpit - select the three dots next to the service key/binding and then [Download](#).
- Use the following CF command extract the data from the service key:

```
cf service-key <my-service> <my-key>
```

Example of a sample service key or binding object:

```
{
  "uaa": {
    "apiurl": "https://api.authentication.<cf_domain>",
    "certificate": "<x509_certificate>",
    "certificate-pinning": true,
    "certurl": "https://<subdomain>.authentication.cert.<cf_domain>",
    "clientid": "<client_id>",
    "clientx509enabled": true,
    "credential-type": "x509",
    "identityzone": "<identitiy_zone>",
    "identityzoneid": "<identitiy_zone_id>",
    "key": "<private_key>",
    "sburl": "https://internal-xsuaa.authentication.<cf_domain>",
    "subaccountid": "<subaccount_id>",
    "tenantid": "<tenant_id>",
    "tenantmode": "dedicated",
    "uaadomain": "authentication.<cf_domain>",
    "url": "https://<subdomain>.authentication.<cf_domain>",
    "verificationkey": "<verification_key>",
    "xsappname": "<xsappname>",
    "zoneid": "<zone_id>"
  },
  "url": "https://api.auditlog.<cf_domain>:6081",
  "vendor": "SAP"
}
}
```

- Obtaining credentials for using the Audit Log Write API

The service key/binding information you retrieve in the previous step, contains the issued client certificates for authentication and the URLs to query the service. You need to use those certificate credentials to issue a JWT token for authentication towards the Write API.

To prepare a request for issuing a JWT token, extract the PEM certificates and the relevant service data. Extract clientid, certurl, certificate and key from the credentials for the audit log service instance. Save the extracted certificate and key to files in .pem format by removing all \n entries.

You can use the following example, considering that the service key/binding information is saved to a json file and use the cat and jq commands can be used for parsing the file:

```
cat customer-write-api-demo-sk.json | jq --raw-output .uaa.certurl
cat customer-write-api-demo-sk.json | jq --raw-output .uaa.clientid
cat customer-write-api-demo-sk.json | jq --raw-output .url
cat customer-write-api-demo-sk.json | jq --raw-output .uaa.certificate > mtls-
certificate.pem
cat customer-write-api-demo-sk.json | jq --raw-output .uaa.key > mtls-private-
key.pem
```

Then execute a POST request to <uaa.certurl>/oauth/token using the client certificates. Use the following parameters in the request:

```
URL: <value of "uaa.certurl">/oauth/token?
grant_type=client_credentials&client_id=<value of "uaa.clientid">
Method: POST
Authentication Method: Client Certificates
Certificate: <Value of "uaa.certificate">
Key: <Value of "uaa.key">
grant_type: client_credentials
client_id: <Value of "uaa.clientid">
```

Example of such a request using curl:

```
curl --cert <path to certificate.pem> --key <path to key.pem>
--request POST https://<value of "uaa.certurl">/oauth/token -d
'grant_type=client_credentials&client_id=<Value of "uaa.clientid">'
```

You can also execute the JWT token request via REST client, like Postman. In such case, make sure to configure the certificates in the Postman settings as described at <https://learning.postman.com/docs/sending-requests/certificates/>.

Pass the issued JWT token in the Authorization header when calling the Audit Log Server Write REST API:  
Authorization: Bearer <token-value>. Example:

```
curl --location --request POST 'https://api.auditlog.<url-host>:6081/audit-log/oauth2/v2/configuration-changes' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer <token>' \
--data-raw '{message-content}'
```

For more examples, see section [Example requests for the Audit Log Write API](#).

The JWT token has a limited validity. When the token validity expires, you can issue a new one with the same set of certificate credentials.

## Rotate credentials

Once the validity of the certificate credentials from the service key and service binding expire, the credentials for using the Audit Log service become invalid until the certificates are rotated. Make sure to properly rotate the credentials before they expire to avoid issues using the Audit Log service.

- For the case of CF apps with service bindings, do the credentials rotation by recreating the service binding with the auditlog service instance (`cf unbind-service`; `cf bind-service`). On each bind operation a new set of x509 certificates is generated.
- For the case of apps using auditlog credentials from an auditlog service key, rotate the credentials by creating a new service key and extracting the certificates and private key from the newly created key.

For security reasons, don't use very long validity period for the mTLS credentials and rotate the certificates as frequently as possible (depending on the use case and application lifecycle).

## Audit Log Write REST API specifics

### Endpoints

The endpoints exposed for the [premium](#) service plan for the 4 types of audit log categories are the following, where the <host> can be obtained from the service key/service binding:

- `https://api.auditlog.<host>:6081/audit-log/oauth2/v2/security-events`
- `https://api.auditlog.<host>:6081/audit-log/oauth2/v2/configuration-changes`
- `https://api.auditlog.<host>:6081/audit-log/oauth2/v2/data-accesses`
- `https://api.auditlog.<host>:6081/audit-log/oauth2/v2/data-modifications`

### Response Status Codes

The following response status codes can be returned from the Audit Log Write API:

Code	Description
201 (Created)	Audit log successfully created.

Code	Description
400 (Bad Request)	Bad request for one of the following reasons: <ul style="list-style-type: none"> <li>• URI syntax error</li> <li>• Invalid attributes</li> <li>• Cannot parse the received audit log message</li> <li>• Connection shutdown</li> </ul>
401 (Unauthorized)	Missing or invalid <i>Authorization</i> header.
413 (Payload Too Large)	The Audit log message size exceeds 10KB.
429 (Too Many Requests)	Rate limit per tenant (provider_tenant, org, space) is hit.
500 (Internal Server Error)	An unexpected exception occurred.
502 (Bad Gateway)	Load balancer issue.
503 (Service Unavailable)	Global rate limit is hit.
504 (Gateway Timeout)	Load balancer issue.

## Audit Log Message Format

The audit log record contains the following two main parts - the first one is filled in by the user of the audit log service and the second one, mainly in the metadata part of the message, is filled in by the audit logging framework.

- **Filled by the audit log user**

Field	Semantics
uuid	Unique identifier of the message.
time	The time when the auditable event has been executed.
user	The user that is logged in and performs the auditable action. To let the server-side extract the user from the JWT token, set the "\$USER" string, or set a custom user and set a custom field.
data_subject	The persona that is the owner of the personal data, which is being accessed or modified in Log read access to sensitive personal data events and Log changes to personal data.
object	The unique identifier for the location of the personal data that is being accessed or modified in case of Log read access to sensitive personal data events and Log changes to personal data, as well as the unique identifier of the location and configuration that is being changed.
attributes	The list of attributes that are either added, deleted, or modified. There are three scenarios, for example: <ol style="list-style-type: none"> <li>1. a new address is added - only fill in the new value for the attribute with the newly written value (skip the old value);</li> <li>2. an existing address is modified – add the changed value as new and the previous value as old;</li> <li>3. an existing address is deleted - fill in only the deleted value as the old value for the attribute (skip the new value field).</li> </ol>
attachment	The list of attachments that contain a personal data and has been opened for reading by a user, and which must be reflected in an audit log entry.

Field	Semantics
ip	The IP where the audited security-relevant event is executed (only available for security audit log events).
data	The message that contains the needed information for the logged security event (only available for security audit log events).
tenant	The identifier of the owner of the audit log message. Each audit log entry can have only single tenant equal to the zone ID, which determines the subaccount that is the owner of the message. The message is visible only for users in a role that has permissions to read the audit logs for this subaccount. The retention of the audit log message is determined based on the setup retention for this subaccount. For the premium plan, set the string "\$PROVIDER", which lets the servers extract the tenant zone ID automatically from the used JWT token.
success	The field is used for transactional messages only. When you mean to log the action as intention, skip the success field. If with the audit log message you mean to confirm the success of the action execution or to state that it was not successful, add the "success" field with values TRUE or FALSE.

- **Filled by the framework**

Field	Semantics
service_instance_id	The GUID of the audit log instance that is used to write audit logs.
platform	The platform identifier together with the region information from where the audit log is written.
org_name	Empty in the newest version for security reasons.
org_guid	The organization GUID of the component that is using the audit log service to write audit logs.
space_name	Empty in the newest version for security reasons.
space_guid	The space GUID of the component that is using the audit log service to write audit logs.
app_name	Empty in the newest version for security reasons.
app_guid	The GUID of the application that is using the audit log service to write audit logs.
category	Message category is set implicitly by the framework.

## Example Requests for the Audit Log Write API

This section contain examples for writing audit log events using curl demo purposes. You can use any other client, the examples serve to demonstrate the structure of the required payload per each audit log type.

### Writing a Security Event

Mandatory fields: `uuid`, `user`, `time`, `data`, `tenant`.

```
curl --location --request POST 'https://api.auditlog.cf.<host>:6081/audit-log/oauth2/v2/security-events' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer <JWT-token-value>' \
--data-raw '{
  "uuid": "<uuid-value>",
  "user": "$USER",
```

```

    "time": "2023-06-30T00:00:00.000Z",
    "ip": "127.0.0.1",
    "data": "Demo security event message.",
    "tenant": "$PROVIDER"
}

```

## Writing a Configuration Change Event

Mandatory fields: object\_id, uuid, user, tenant, time, attributes.

```

curl --location --request POST 'https://api.auditlog.cf.<host>:6081/audit-log/
oauth2/v2/configuration-changes' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer <JWT-token-value>' \
--data-raw '{
    "uuid": "<uuid-value>",
    "user": "test-als-user@sap.com",
    "time": "2023-06-30T00:00:00.000Z",
    "id": "<id-value>",
    "tenant": "$PROVIDER",
    "object": {
        "type": "name of an attribute",
        "id": {
            "key1": "value1",
            "key2": "value2"
        }
    },
    "attributes": [
        {
            "name": "cfg attribute 1",
            "old": "old cfg attribute 1 value",
            "new": "new cfg attribute 1 value"
        },
        {
            "name": "cfg attribute 2",
            "old": "old cfg attribute 2 value",
            "new": "new cfg attribute 2 value"
        }
    ],
    "category": "audit.configuration",
    "customDetails": {
        "customKey": "customValue",
        "arbitraryKey": "arbitraryValue",
        "customJson": {
            "testString1": "testValue1",
            "testString2": "testValue2"
        }
    }
}

```

## Writing a Data Modification Event

Mandatory fields: object\_id, user, tenant, time, attributes.

```

curl --location --request POST 'https://api.auditlog.cf.<host>:6081/audit-log/
oauth2/v2/data-accesses' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer <JWT-token-value>' \
--data-raw '{
    "uuid": "<uuid-value>",
    "user": "some-user-id",
    "identityProvider": "$IDP",
    "time": "2023-06-30T00:00:00.000Z",
    "tenant": "$PROVIDER",
    "object": {
        "type": "online system",
        "id": "some-object-id"
    }
}

```

```

        "id": {
            "name" : "Students info system",
            "module" : "Admin module"
        }
    },
    "data_subject": {
        "type": "student",
        "role": "foreign student",
        "id": {
            "student_id": "12345678",
            "first name": "Jane",
            "last name": "Smith"
        }
    },
    "attributes": [
        {
            "name": "name of attribute 1"
        },
        {
            "name": "name of attribute 2",
            "successfull": true
        }
    ],
    "customDetails" : {
        "customKey" : "customValue",
        "anotherCustomKey" : "another custom value",
        "customJson": {
            "testString1": "testValue1",
            "testString2": "testValue2"
        }
    },
    "attachments": [
        {
            "id": "someId",
            "name": "attachmentName",
            "content": "attachmentContent"
        }
    ]
}

```

## Writing a Data Access Event

Mandatory fields: object\_id, user, tenant, time, attributes.

```

curl --location --request POST 'https://api.auditlog.cf.<host>:6081/audit-log/
oauth2/v2/data-modifications' \
--header 'Content-Type: application/json' \
--header 'Authorization: Bearer <JWT-token-value>' \
--data-raw '{
    "uuid": "<uuid-value>",
    "user": "$USER",
    "time": "2023-06-30T00:00:00.000Z",
    "object": {
        "type": "DEMO Instance",
        "id": {
            "serviceInstanceId": "<serviceInstanceId-value>"
        }
    },
    "data_subject": {
        "type": "credentials",
        "role": "authentication",
        "id": {
            "key1": "value1",
            "key2": "value2"
        }
    },
    "attributes": [

```

```

{
  "name": "some attribute",
  "new": "new value"
}
],
"id": "<id-value>",
"category": "audit.data-modification",
"tenant": "$PROVIDER",
"customDetails": {}
}'
```

## Getting Support

If you need support for Audit Log Write API, open a SNOW support ticket to the component BC-CP-CF-SEC-AUDITLG. For more information, see Related information.

## Related Information

[Getting Support \[page 3148\]](#)

### 6.2.7.4.1 Categories of Audit Relevant Events

Find explained the four categories of Audit Relevant Events.

Category	Description	Examples
Log read access to sensitive personal data	Use a data-accesses audit log record for the cases where an access to personal data is executed. The audit log contains information on the accessed personal data, as well as the full information of the user that is accessing the personal data, the owner of the personal data (the Data Subject), and the identifier of the place where the access to the personal data have been executed (the Object).	Examples of personal data access events are all the read accesses sensitive personal data, which means information on racial or ethnic origin, political opinions, religious or philosophical beliefs, trade-union membership, health or sex life, bank account and credit card data, genetic data and biometric data for the purpose of uniquely identifying a natural person.
Log changes to personal data	Use a data-modification audit log record for the events that are related to modification of personal data. The audit log contains information on the modified personal data, as well as the full information of the user that is accessing the personal data, the owner of the personal data (the Data Subject), and the identifier of the place where the access to the personal data have been executed (the Object).	Examples of personal data modification events are writing, deleting, or changes on an address, e-mail address, phone number, and any other data that can lead to a persona identification.

Category	Description	Examples
Security event log	Log security data for all events that might have an impact over the system security.	Examples for such events are successful and non-successful login and logout events, changes in access control permissions, changes in credentials, changes in sensitive configurations, and so on.
Configuration change log	Use the configuration change audit log events for various changes in configurations. The audit log events contain the changed configuration parts with old and new value. Log only the changed parts with the old and new value. If old or new value doesn't exist in case of adding a new property or deleting one respectively, skip the missing values or set them as null.	Examples of configuration change instances, binding/unbinding service instances, software update, changes in configurations, and so on.

## 6.2.7.5 SAP Audit Log Viewer service for the Cloud Foundry Environment

The SAP Audit Log Viewer service displays the audit logs for your Cloud Foundry account, produced by SAP applications and services you've subscribed to.

The UI allows you to:

- Read the audit logs written for your account in selected time frame. The SAP Audit Log Viewer service displays 500 audit log records per request. If you want to view more audit log records, specify a shorter time frame.
- Display more detailed information on each single audit log record.
- Filter the retrieved client-side chunk for certain strings.
- Select which of the four audit log message categories to be returned. If you don't select any category, all are returned.
- Download the audit logs in the selected time frame.

The default displayed columns are:

- User - the person that has executed the event reflected in the written audit log.
- Time - creation time for the audit log message.
- Message - summary of the audit log message that is written.
- Category - audit log message category. The four-supported audit log message categories are:
  - audit.security-events
  - audit.configuration
  - audit.data-access
  - audit.data-modification

The appearance of the UI can be modified by selecting the rows to be displayed on a single page, as well as the columns that you want to be visible.

## 6.2.7.5.1 Subscribe to SAP Audit Log Viewer service

Find explained the subscription procedure to the to the Audit Log Viewer service.

### Prerequisites

Before subscribing to the Audit Log Viewer service, you need to configure the relevant entitlement in your Global Account. For more information, see **Related Information**.

### Procedure

1. To subscribe to SAP Audit Log Viewer service, use the SAP BTP cockpit and go to the  [Services](#)  [Instances and Subscriptions](#)  [Subscriptions](#)  tab of your subaccount.
2. Once you're subscribed, select [Go to Application](#) to open the SAP Audit Log Viewer service and log in there.

### Related Information

[Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#)

## 6.2.7.5.2 Access Audit Log Viewer service

### Context

To retrieve the audit logs for your subaccount using the Audit Log Viewer service, you need to have proper authorizations. See <https://docs.cloudfoundry.org/concepts/roles.html#permissionsInformation> ↗, published on a non-SAP site.

### Procedure

1. Create a RoleCollection.
2. Search for roles with the name "Auditlog\_Auditor" and select both entries with the following application identifiers:

- auditlog-management!b\*
  - auditlog-viewer!t\*
3. Assign the role to a user or create a rule to assign it to users based on the SAML Assertion coming from the IDP.

**ⓘ Note**

Only account members with the Security Administrator role are authorized to edit application authorizations.

### 6.2.7.5.3 Accessibility Features in Audit Log Viewer

To optimize your experience of Audit Log Viewer, SAP Business Technology Platform (SAP BTP) provides features and settings that help you use the software efficiently.

**ⓘ Note**

Audit Log Viewer is based on SAPUI5. For this reason, accessibility features for SAPUI5 also apply. See the accessibility documentation for SAPUI5 on SAP Help Portal at [Accessibility for End Users](#).

### 6.2.7.6 Security Events Logged by the Services

This document contains a list of the Cloud Foundry services that log security events. The logged events are service specific and determined by the service itself. The Audit Log service does not know what security events the services log, it only provides the infrastructure for services to write the logs, and for customers to read them by using the Audit Log Retrieval API and the Audit Log Viewer. For more information about the security events logged by a service, follow the corresponding link in the table.

Audit Log Events by Service

Service	Audit log events page of the service
Authorization and Trust Management	<a href="#">Auditing and Logging Information for SAP Authorization and Trust Management Service</a>
Business Entity Recognition	<a href="#">Auditing and Logging Information</a>
Business Logging	<a href="#">Auditing and Logging</a>
Business Rules	<a href="#">Auditing and Logging Information</a>
Custom Domain (Cloud Foundry)	<a href="#">Auditing and Logging Information</a>
Data Attribute Recommendation	<a href="#">Auditing and Logging Information</a>
Document Classification	<a href="#">Auditing and Logging Information</a>
Document Information Extraction	<a href="#">Auditing and Logging Information</a>
Document Management	<a href="#">Auditing and Logging Information</a>

Service	Audit log events page of the service
Identity Authentication	Auditing and Logging Information
Invoice Object Recommendation	Auditing and Logging Information
Process Visibility	Auditing and Logging Information
SAP AI Core	Auditing and Logging Information
SAP AI Launchpad	Auditing and Logging Information
SAP Alert Notification Service	Auditing and Logging Information
SAP API Management	Auditing and Logging Information
SAP Build Apps	Auditing and Logging Information
SAP Build Process Automation	Audit Logging
SAP Build Work Zone, standard edition	Auditing and Logging Information
SAP Build Work Zone, advanced edition	Auditing and Logging Information
SAP Business Application Studio	Auditing and Logging Information
SAP Cloud Identity Access Governance	Integration of Audit Log Service
SAP Cloud Integration (Cloud Foundry)	Auditing and Logging Information
SAP Cloud Integration for data services	Security Log
SAP Cloud Portal service	Auditing and Logging Information
SAP Cloud Transport Management	Auditing and Logging Information
SAP Credential Store	Auditing and Logging Information
SAP Conversational AI	Auditing and Logging Information
SAP Data Privacy Integration	Access to Audit Logs
SAP Datasphere Cloud	Audit Logging
SAP Enterprise Architecture Designer, Cloud Edition	Auditing and Logging
SAP Edge Services, Cloud Edition	Auditing and Logging Information
SAP Event Mesh	Auditing and Logging Information
SAP HANA Service for SAP BTP in AWS and Google Cloud Regions	Auditing
SAP HANA Service for SAP BTP in SAP Regions	Auditing
SAP Integration Advisor	Auditing and Logging Information
SAP Leonardo Machine Learning Foundation	Logging
SAP Market Rates Management, Bring Your Own Rates data option	Auditing and Logging Information
SAP Market Rates Management, Refinitiv data option	Auditing and Logging Information
SAP Mobile Services (Cloud Foundry)	Viewing Audit Logs
SAP Monitoring Service	Auditing and Logging Information
SAP Omnichannel Promotion Pricing	Auditing and Logging Information

Service	Audit log events page of the service
SAP Print Service	<a href="#">Auditing and Logging Information</a>
SAP Private Link Service (Beta)	<a href="#">Audit and Logging Information</a>
SAP S/4HANA Cloud Extensibility	<a href="#">Auditing and Logging Information</a>
SAP Secure Login Service for SAP GUI	<a href="#">Auditing and Logging Information</a>
SAP Start	<a href="#">Security Events Logged by the Interest Cards Service</a>
SAP SuccessFactors Extensibility	<a href="#">Auditing and Logging Information</a>
SAP Task Center	<a href="#">Auditing and Logging Information</a>
SAP Variant Configuration and Pricing	<a href="#">Audit Logging</a>
SAP Web Analytics	<a href="#">Auditing and Logging Information</a>
SAP Web IDE Full-Stack	<a href="#">Auditing and Logging Information</a>
Service Ticket Intelligence	<a href="#">Auditing and Logging Information</a>
UI Theme Designer	<a href="#">Auditing and Logging Information</a>
UI5 Flexibility for Key Users	<a href="#">Auditing and Logging Information</a>
Unit of Measure	<a href="#">Audit Logging</a>
Workflow	<a href="#">Audit Logs</a>

## 6.2.7.7 Checking Who Created an Application or a Service Instance

When you want to know for your subaccount which user created an application or a service instance, you need to look into the event logs.

- [Check Who Created an Application \[page 2532\]](#)
- [Check Who Created a Service Instance \[page 2534\]](#)

### 6.2.7.7.1 Check Who Created an Application

Creating an application is logged by SAP Audit Log service. You can see this information in the Audit Log Viewer or retrieve it by the Audit Log API.

Example in Audit Log Viewer for SAP Business Application Studio created by <user\_email> on 16.03.2023 08:44:06.

Log Entry Details	
Timestamp	16 März 2023, 09:44:06.321 +0100
Log Message	Configuration modification message. Attribute with name "com.sap.core.commercial.service.registry.models.RootSubscription" was changed from "" to "[{"id":3706115,"propertyDataSet":[],"internalSubscriptionId":"us10-app-studio!t5804_f923737b-627b-4024-bfee-d028c0483b27_us10-app-studio!t5804","xsuaaSaaSApplicationServiceInstanceId":"d752ae3d-b04a-4754-b6b0-f9b787bf0fe7e","rootIdentifier":"us10-app-studio!t5804","appId":"us10-app-studio!t5804","appName":"sapappstudiotrial","consumerId":"f923737b-627b-4024-bfee-d028c0483b27","subaccountId":"f923737b-627b-4024-bfee-d028c0483b27","subdomain": "test-cyj738uy","globalAccountID":"c3cf7de-a664-40b4-b3cb-d480785c08f9","url":null,"code": "trial","amount":1,"state": "IN_PROCESS","stateDetails":null,"appError":null,"origin": "CONSUMER"}]". The attribute is part of an object with type "CREATE" and id consisting of: entity "RootSubscription : { xsuaaAppId: us10-app-studio!t5804, consumerTenant: f923737b-627b-4024-bfee-d028c0483b27, subdomain: test-cyj738uy }".
IP	[REDACTED]
User	[REDACTED]
Category	audit.configuration
Data	<p>Created by user and timestamp</p> <p>Type of action: CREATE</p> <p>First indication of service name</p> <p>Service ID (little bit obscured)</p> <pre> 12 -   "message": { 13 -     "uid": "0a45fec4-e2a8-4c35-9815-c2e3a696dbff", 14 -     "user": "&lt;user_email&gt;", 15 -     "time": "2023-03-16T08:44:06.321Z", 16 -     "id": "4e31b6b1-c73e-4491-93d2-a6790a13ed75", 17 -     "success": true, 18 -     "object": { 19 -       "type": "CREATE", 20 -       "id": { 21 -         "entity": "RootSubscription : { xsuaaAppId: us10-app-studio!t5804, consumerTenant: f923737b-627b-4024-bfee-d028c0483b27, subdomain: test-cyj738uy }" 22 -       } 23 -     }, 24 -     "attributes": [ 25 -       { 26 -         "name": "com.sap.core.commercial.service.registry.models.RootSubscription", 27 -         "old": "", 28 -         "new": "[{"id":3706115,"propertyDataSet":[],"internalSubscriptionId":"us10-app-studio!t5804_f923737b-627b-4024-bfee-d028c0483b27_us10-app-studio!t5804","xsuaaSaaSApplicationServiceInstanceId":"d752ae3d-b04a-4754-b6b0-f9b787bf0fe7e","rootIdentifier":"us10-app-studio!t5804","appId":"us10-app-studio!t5804","appName":"sapappstudiotrial","consumerId":"f923737b-627b-4024-bfee-d028c0483b27","subaccountId":"f923737b-627b-4024-bfee-d028c0483b27","subdomain": "test-cyj738uy","globalAccountID":"c3cf7de-a664-40b4-b3cb-d480785c08f9","url":null,"code": "trial","amount":1,"state": "IN_PROCESS","stateDetails":null,"appError":null,"origin": "CONSUMER"}]" 29 -     } 30 -   } 31 -   "status": "END" </pre>

The same information retrieved by Audit Log API:

Trial Audit Log API / GET AUDITLOG

Save

Send

GET https://auditlog-management.cfapps.us10.hana.ondemand.com/auditlog/v2/auditlogrecords?time\_from=2023-03-16T07:00:00&time\_to=2023-03-16T14:00:00

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
time_from	2023-03-16T07:00:00	
time_to	2023-03-16T14:00:00	

Body Cookies Headers (14) Test Results

Status: 200 OK Time: 665 ms Size: 97.79 KB Save Response

Pretty Raw Preview Visualize JSON

```
549 "message_uuid": "ae45fec4-e2a8-4c35-9815-c2e3e696dbff",
550 "time": "2023-03-16T08:44:06.321Z",
551 "tenant": "f923737b-627b-4024-bfee-d028c0483b27",
552 "org_id": "92f1da92-e5b3-4cc5-8c90-964165af11c8",
553 "space_id": "92f1da92-e5b3-4cc5-8c90-964165af11c8",
554 "app_or_service_id": "92f1da92-e5b3-4cc5-8c90-964165af11c8",
555 "als_service_id": "e4d10a91-aa3c-4c30-8051-a767925e0de1",
556 "user": "<user_email>",
557 "category": "audit.configuration",
558 "format_version": "",
559 "message": {
    "id": "\\\\"e431b0b1-c73e-4491-93d2-a6790e13ed75\\\"",
    "success": true,
    "object": {
        "type": "CREATE",
        "id": "\\\\"entity\\\"",
        "RootSubscription": {
            "xsuaaAppId": "us10-app-studio:t5804",
            "consumerTenant": "f923737b-627b-4024-bfee-d028c0483b27",
            "subdomain": "test-cy7380uy"
        },
        "attributes": [
            {
                "name": "com.sap.core.commercial.service.registry.models.RootSubscription"
            }
        ],
        "internalSubscriptionId": "\\\\"us10-app-studio:t5804_f923737b-627b-4024-bfee-d028c0483b27\\\"",
        "xsuaaSasApplicationServiceInstanceId": "\\\\"d72ea3d-h04a-475d-b60f-0b797b0fe7e\\\"",
        "xsuaaSasApplicationServiceInstanceId": null,
        "rootIdentifier": "\\\\"us10-app-studio:t5804\\\"",
        "appId": "\\\\"us10-app-studio:t5804\\\"",
        "appName": "\\\\"sapappstudiotrial\\\"",
        "consumerId": "\\\\"f923737b-627b-4024-bfee-d028c0483b27\\\"",
        "subaccountId": "\\\\"f923737b-627b-4024-bfee-d028c0483b27\\\"",
        "subdomain": "\\\\"test-cy7380uy\\\"",
        "globalAccountId": "\\\\"3c3cf7de-a664-40d4-b3c3-0480785c08f9\\\"",
        "url": "\\\\"/\\\"",
        "code": "\\\\"trial\\\"",
        "amount": 1,
        "state": "\\\\"IN_PROCESS\\\"",
        "details": "\\\\"error\\\\\\\";\\\"CONSUMER\\\\\\\";\\\"original\\\\\\\";\\\"status\\\\\\\";\\\"EMD\\\\\\\";\\\"category\\\\\\\";\\\"audit.configuration\\\\\\\";\\\"tenant\\\\\\\";\\\"f923737b-627b-4024-bfee-d028c0483b27\\\"\\\"",
        "customDetails": "\\\\"{}\\\""
    }
}
```

## 6.2.7.7.2 Check Who Created a Service Instance

The following steps help you get information about who created a service instance. These steps are valid only for Cloud Foundry subaccounts.

To get the information for a service instance, use the [Cloud Foundry API](#), which is available for all Cloud Foundry subaccounts. The following steps describe a path of API calls. Combining all information from the three steps, you get a list of all services created by a specific user at a specific time.

1. Get a list of all audit events from type audit.service\_instance.create.

Trial Audit Log API / GET CF Audit Events

The screenshot shows a POST request to `https://api.cf.us-10-01.hana.ondemand.com/v3/audit_events?page=1&per_page=50&types=audit.service_instance.create`. The response is a JSON object with the following structure:

```
1 {
2   "pagination": {
3     "total_results": 6, Related to current 6 instances in BTP cockpit
4     "total_pages": 1,
5     "first": { ... },
6     "last": { ... },
7     "next": null,
8     "previous": null
9   },
10   "resources": [
11     { ... },
12     { ... },
13     { ... },
14     { ... },
15     { ... },
16     { ... }
17   ]
18 }
```

One resource is highlighted with a red box:

```
155   {
156     "guid": "10f88ff85-e4e0-4446-823d-d6ddb8b05e63",
157     "created_at": "2023-03-16T08:46:02Z", Timestamp
158     "updated_at": "2023-03-16T08:46:02Z",
159     "type": "audit.service_instance.create",
160     "actor": {
161       "guid": "289dd350-f1f7-4450-884f-9680b1e479b5",
162       "type": "user",
163       "name": "<user_email>" Creator
164     }
165   }
```

Another resource is highlighted with a red box:

```
167   "data": {
168     "request": {
169       "name": "alertnotificationinstance" Name of the instance. Not the service itself!
170       "type": "managed",
171       "relationships": {
172         "space": {
173           "data": {
174             "guid": "b9c9b7e4-8f1c-484a-b1a7-29374b0d7fa6"
175           }
176         },
177         "service_plan": {
178           "data": {
179             "guid": "7a3005c2-6545-4e48-bed0-a2b7096d81e6" Used to navigate to service plan information
180           }
181         }
182       }
183     }
184   }
```

2. Follow the link to the service plan.

GET https://api.cf.us10-001.hana.ondemand.com/v3/service\_plans/7a3005c2-6545-4e48-bed0-a2b7096d81e6

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Body Cookies Headers (19) Test Results

Pretty Raw Preview Visualize JSON

```

1   "guid": "7a3005c2-6545-4e48-bed0-a2b7096d81e6",
2   "created_at": "2021-04-07T06:36:19Z",
3   "updated_at": "2023-03-17T12:59:25Z",
4   "name": "standard",
5   "visibility_type": "organization",
6   "available": true,
7   "free": false,
8   "costs": [],
9
10  "description": "Allows consumption of SAP Alert Notification service events as well as posting custom events",
11  "maintenance_info": {},
12  "broker_catalog": {...},
13  },
14  "schemas": {...},
15  },
16  "relationships": {...},
17  },
18  "metadata": {...},
19  },
20  "links": {
21    "self": {...},
22    "service_offering": {
23      "href": "https://api.cf.us10-001.hana.ondemand.com/v3/service_offerings/261c7e51-90e2-4718-82ed-0850ca45996f"
24    },
25    "visibility": {...}
26  }
27}
28

```

**First indication of name, but not reliable.**

**Navigation to service offering where we can find the service name (and SAP internal SERVICE-ID)**

- See the service offering parameters to get the name, documentation URL and other parameters of the service.

GET https://api.cf.us10-001.hana.ondemand.com/v3/service\_offerings/261c7e51-90e2-4718-82ed-0850ca45996f

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Body Cookies Headers (19) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```

z2ZC7wKQsK/iu84Al38hNDFCr5e0CB24Z/fRfZeVbscnhgS6VoE+ppp5xGVFNm5lJtgWt16SQumaoLTETdehgbf00TGhA/fuHt52Selp5eNcpKuvuDogZI4P3kWrrJpwReDhtCcp
+xF5UzDw3dTQU/TbsFDsBQR5j3RkXe0ToudAbq49+SY/m1Zqc7Qah7NerjxGy30JY55MM9fCcEdkE3GL9Fa9c7juuEdsC5bGfVnpyI+JZFe4FAtA/
1svVsN6pmh4e2eb4hBXLhtCuVe1BgRaYvAAAR51TehgypCgWUDesCc94tsI0d0RL42LB9E3dFwoY+tZNC0e81K797ePhkekbgDR5MRKNd4ucentAZP1qvpykM9jd6uErTPKII
+u0ct0jaekwEBNCyZq1Sz+SnUBpjRoTqH7xfhSqe07X1lF1H1ygaRSjUi1kb5tTCkZ33vIC/5CpAEE10C8GewcqotpflVG/2EBxFCh6wAg42APForgI1wpdx8mGGLMwLIdC0W4U
+uFdmzfC3jzOaKn18qovFH4UGRuD00IE31g9NuFijjIF+H4YEV5Vsqqqf/g/v2W65dhvEHNACy239EuIJstcIC5KC5xFyqyhzhbhK1Do+rQITqdF506PjTZTXiymh+XCCI+OkPgA+
IRU4Riy3y53P07Qr46E1NBNTB1psopkxMRLRvYkojic30a1hSecZNJwspk8kQJHYCc+6jSm2333p0yAOTU4cD1kMznqkD06sQJXqd0o2pXWkFcfsk1DYAbnov8vScjcy1VboU
+KFdkAq7jIKpqjgnX24ev5coovvCPJ45u4K/iMjptwbsIEK3HT45ug61MKbR9+D7Nyjgrce3h6HOBf13jaZo114suDpHtzbef7o01B0M66EU0500iuZcM9jbDV02T
+wDdc1wG3foI1JktINGOLjk3aElu5uq5CKPkgXvuL3w91LAw2rtoCwrajdHvUTPxu0BFc30hcB+vjwteJJD3vuAKMwQ0dxSMzRE1uKQIW+BLChz2lCqt54eG3a4ru1Ag+EHwKk0
01NZBodzfdXGsdZuRCNSUNFJOM0IVuk0guCeybFxSmbMsA2rDw8C+CxoFl+oMYYeBiylTik1Vik1EKsXAbb5pW41m1xG1qEUnB1rXUFGsg9P8BcsuyGVIIG+4AAAASUVORK5C
16  "supportUrl": "https://help.sap.com/viewer/p/ALERT_NOTIFICATION",
17  "displayName": "Alert Notification",
18  "documentationUrl": "https://help.sap.com/viewer/p/ALERT_NOTIFICATION",
19  "serviceInventoryId": "SERVICE-334",
20  "sap": {
21    "tenant_aware": true,
22    "instance_isolation": true
23  },
24  "sm_offering_id": "6f3240fe-c9f9-4b7e-b8a5-7528269106a2",
25  },
26  "features": {
27    "plan_updateable": true,
28    "bindable": true,
29    "instances_retrievable": false,
30    "bindings_retrievable": false,
31    "allow_context_updates": false
32  },
33}

```

## 6.2.7.8 Troubleshooting

You can use the [Guided Answers for Audit Logging in the Cloud Foundry Environment](#) page as self service to troubleshoot your issues.

If you don't manage to solve your issue with the Guided Answers, you can open a support ticket in [SAP4Me](#), in which you need to provide:

- Your GA ID
- Your subaccount ID
- Step by step description of the issue with screenshots.

## 6.3 Administration and Operations in the ABAP Environment

As an administrator in the ABAP environment, you have different tools and features available that support you in your daily work.

### Task Examples

As an administrator in the ABAP environment, you focus on administration and operational tasks that aren't managed by SAP. Such tasks can include, for example:

- Account and organization management, including identity and access management for your users
- Monitoring of performance issues and exceptions in your own applications  
Such a monitoring includes response times of application-related communication endpoints, SQL statements issued by the application, application jobs, or ABAP runtime errors originating from custom code.
- System sizing, including a regular review of the overall resource utilization  
Such a system sizing can include a daily check of resource utilization peaks and a monthly check of resource utilization trends.
- Software lifecycle of custom applications (add-on updates or custom transports)
- Tenant management if you manage multi-tenant custom applications
- Monitoring of integrations with other cloud or on-premise based services

## Tools, Apps, and Functions: Overview

Here's an overview of tools, apps, and functions that are available to you to perform administration and operations tasks, such as the following:

Task	Tools, Apps, or Functions
Managing accounts and organizations	<p>As an administrator in the ABAP environment, you can create additional accounts for users. In addition, you can manage orgs, spaces, and space quota plans to organize your subaccount into smaller units. You use the tools for account and org administration from the Cloud Foundry environment for this purpose.</p> <p>For more information, see the following:</p> <ul style="list-style-type: none"><li>• <a href="#">Account Administration in the Cockpit</a></li><li>• <a href="#">Org Administration Using the Cockpit</a></li><li>• <a href="#">Org Administration Using the Cloud Foundry CLI</a></li></ul> <p>For more information about business roles in the ABAP environment, see <a href="#">Business Catalogs and Business Roles [page 2539]</a>.</p>
Performing administration tasks, such as identity and access management, business configuration, communication management, and more	In the Fiori launchpad, you can find apps that support you as an administrator or developer with various operational tasks. For more information, see <a href="#">SAP Fiori Apps in the ABAP Environment [page 2538]</a> .
Performing technical operations, such as monitoring the performance of your systems as well as the exhaustion of your purchased service volume	You can use tools such as the technical monitoring cockpit, health monitoring, and more. There's also guidance available for the most common tasks in technical operations and for system sizing. For more information, see <a href="#">Technical Operations [page 2926]</a> .
Operating communications when you integrate your system or solution with other systems to enable data exchange in your ABAP environment	For more information, see <a href="#">Communication Operations [page 2843]</a> .
Performing lifecycle management operations and system administration tasks such as <ul style="list-style-type: none"><li>• stopping currently unused systems and scheduling (regularly recurring) stop/starts for a system to save resource usage.</li><li>• restoring recently deleted tenants that are still in retention time.</li><li>• monitoring operations.</li></ul>	The <a href="#">Landscape Portal</a> functions as a central tool for providers to perform lifecycle management operations and system administration tasks, see <a href="#">Manage System Hibernation, Restore Consumer Tenants, Operations Dashboard</a> .
Getting notifications about planned maintenance events and about unplanned service outages	Register for the Cloud Availability Center. For more information, see the information about the <a href="#">Cloud Availability Center</a>  on SAP Support Portal.

Task	Tools, Apps, or Functions
Operating cloud-centric SAP solution landscapes	<p>SAP Cloud ALM is the central entry point to operate cloud-centric SAP solution landscapes. You can use it for the following:</p> <ul style="list-style-type: none"><li>• Central monitoring of service level indicators (real user monitoring, job monitoring, exception monitoring)</li><li>• Central monitoring of key performance indicators for system and applications (health monitoring)</li><li>• Alert notifications (currently limited to SAP-defined metrics of the health monitoring framework)</li></ul> <p>For more information about the available metrics for health monitoring, see <a href="#">Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM [page 2956]</a>.</p>

### 6.3.1 SAP Fiori Apps in the ABAP Environment

In the SAP Fiori launchpad for the ABAP environment, you can use apps to help you with your tasks as administrator and developer.

Apps for the ABAP environment help you with the following, for example:

- Manage lifecycle maintenance of business users
- Maintain certificate trust lists
- Integrate your systems in the ABAP environment with other systems to enable data exchange
- Manage software components in your system landscape

### Prerequisites

To be able to view and use the apps in the SAP Fiori launchpad, you need a business role with specific business catalogs. You can either use the business roles that are available with the ABAP environment or you can create business roles that include the relevant business catalogs. For more information about the business catalogs and business roles, see [Business Catalogs and Business Roles \[page 2539\]](#). For more information about creating business roles from available business role templates in the ABAP environment, see also [How to Create a Business Role from a Template \[page 2668\]](#).

### Related Information

[SAP Fiori Launchpad - User Guide](#)

## 6.3.1.1 Business Catalogs

### [Business Catalogs and Business Roles \[page 2539\]](#)

Get an overview about which business catalogs and business roles are required for applications in the ABAP environment.

### [Business Catalogs for Development Tasks \[page 2548\]](#)

Get an overview of available business catalogs for development tasks and their restrictions.

### [Business Catalog for Key User Tasks \[page 2550\]](#)

Get an overview of available business role catalogs and their restrictions.

## 6.3.1.1.1 Business Catalogs and Business Roles

Get an overview about which business catalogs and business roles are required for applications in the ABAP environment.

### → Tip

You can also use the *Business Catalogs* app to view the details of each business catalog, including their descriptions and the apps they provide access to.

Area	Application	Business Role Template ID	Business Catalog ID
Extensibility	<a href="#">Custom Logic (Deprecated) [page 2633]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_EXT_BLE <i>Extensibility - Key User Adaptation</i>
Extensibility	<a href="#">Configure Predefined Custom Fields [page 2638]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_EXT_PCF_PC <i>Extensibility - Predefined Custom Fields</i>
Employee Master Data	<a href="#">Maintain Employees [page 2624]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_HCM_BC_EMP_MDT_PC <i>Employee - Master Data</i>
Read Access Logging	<a href="#">Read Access Logging Configuration [page 2733]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_RAL_CONF <i>Read Access Logging - Configuration</i>
Read Access Logging	<a href="#">Read Access Logging: Monitor [page 2756]</a>	SAP_BR_DATA_PRIVACY_SPECIALIST Data Privacy Specialist	SAP_CORE_BC_RAL <i>Read Access Logging - Monitoring</i>

Area	Application	Business Role Template ID	Business Catalog ID
User Interface Configuration	<a href="#">Managing Launchpad Spaces and Pages</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI_FLD <i>User Interface - Fiori Launchpad Design</i>
User Interface Configuration	<a href="#">Manage Launchpad Settings</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI <i>User Interface - Configuration</i>
User Interface Monitoring	<a href="#">Display Launchpad Content Exposure Logs [page 2831]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI_MON <i>User Interface - Monitoring</i>
ABAP Test Cockpit	<a href="#">ABAP Test Cockpit Configurator [page 2553]</a>	SAP_BR_ADMINISTRATOR_S_W_DEV Administrator - Software Development	SAP_CORE_BC_ATC_CONFIG <i>ABAP Test Cockpit Configuration</i>
ABAP Test Cockpit	<a href="#">Manage API Snapshots [page 2556]</a>	SAP_BR_ADMINISTRATOR_S_W_DEV Administrator - Software Development	SAP_CORE_BC_ARS_PC <i>Lifecycle Management - API Snapshots</i>
Business Configuration	<a href="#">Business Configuration Change Logs [page 2572]</a>	SAP_BR_BPC_EXPERT Configuration Expert - Business Process Configuration	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Business Configuration	<a href="#">Manage Number Range Intervals [page 2576]</a>	SAP_BR_BPC_EXPERT Configuration Expert - Business Process Configuration	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Business Configuration	<a href="#">Upload Business Configuration [page 2579]</a>	SAP_BR_BPC_EXPERT Configuration Expert - Business Process Configuration	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Business Configuration	<a href="#">Custom Business Configurations App [page 2574]</a>	SAP_BR_BPC_EXPERT Configuration Expert - Business Process Configuration	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>

Area	Application	Business Role Template ID	Business Catalog ID
Business Configuration	<a href="#">Export Customizing Transports [page 2582]</a>	SAP_BR_BPC_EXPERT <i>Configuration Expert - Business Process Configuration</i>	SAP_CORE_BC_BCT_TRN_MN_G_PC <i>Business Configuration - Transport Management</i>
			SAP_CORE_BC_BCT_TRN_REL_PC <i>Business Configuration - Transport Release Management</i>
Application Jobs	<a href="#">Application Jobs [page 2561]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_APJ_JCE <i>Application Jobs</i>
Application Jobs	<a href="#">Maintain Job Users [page 2569]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_APJ_JCE <i>Application Jobs</i>
Application Jobs	<a href="#">Application Job Templates [page 2566]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_APJ_TPL <i>Application Job Templates</i>
System Management	<a href="#">Maintain User Sessions [page 2779]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_CST_DP_MAS_PC <i>System Management - Sessions</i>
Factory Calendar	<a href="#">Maintain Holidays [page 2641]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CA_BC_IC_LND_CAL_C_A1_PC <i>Factory Calendar - Configuration</i>
Factory Calendar	<a href="#">Maintain Holiday Calendars [page 2643]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CA_BC_IC_LND_CAL_C_A1_PC <i>Factory Calendar - Configuration</i>
Factory Calendar	<a href="#">Maintain Factory Calendars [page 2645]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CA_BC_IC_LND_CAL_C_A1_PC <i>Factory Calendar - Configuration</i>
Health Monitoring	<a href="#">Health Monitoring [page 2786]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_TMC <i>Technical Monitoring Cockpit Infrastructure</i>
Identity and Access Management	<a href="#">Business Catalogs [page 2699]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_IAM_RM <i>Role Management</i>

<b>Area</b>	<b>Application</b>	<b>Business Role Template ID</b>	<b>Business Catalog ID</b>
Identity and Access Management	<a href="#">Business Role Templates [page 2702]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>
Identity and Access Management	<a href="#">Display Authorization Trace [page 2704]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>  SAP_CORE_BC_IAM_RA <i>Role Assignment</i>
Identity and Access Management	<a href="#">Display Restriction Types [page 2685]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>  SAP_CORE_BC_IAM_RA <i>Role Assignment</i>
Identity and Access Management	<a href="#">Display Technical Users [page 2692]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_UM <i>User Management</i>
Identity and Access Management	<a href="#">IAM Information System [page 2701]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>  SAP_CORE_BC_IAM_RA <i>Role Assignment</i>  SAP_CORE_BC_IAM_UM <i>User Management</i>
Identity and Access Management	<a href="#">Maintain Business Roles [page 2666]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RA <i>Role Assignment</i>  SAP_CORE_BC_IAM_RM <i>Role Management</i>
Identity and Access Management	<a href="#">Maintain Business Users [page 2655]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RA <i>Role Assignment</i>  SAP_CORE_BC_IAM_UM <i>User Management</i>

Area	Application	Business Role Template ID	Business Catalog ID
Identity and Access Management	<a href="#">Maintain Business User Groups [page 2660]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	<p><i>User Management</i> (SAP_CORE_BC_IAM_UM)</p> <p><i>Identity and Access Management - User Management - Display</i> (SAP_CORE_BC_IAM_UM_DISP_PC)</p> <p><i>Identity and Access Management - Group Management</i> (SAP_CORE_BC_IAM_GRP_PC)</p>
Identity and Access Management	<a href="#">Maintain Business Role Groups [page 2664]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	<p><i>Role Management</i> (SAP_CORE_BC_IAM_RM)</p> <p><i>Role Assignment</i> (SAP_CORE_BC_IAM_RA)</p> <p><i>Identity and Access Management - Role Management - Display</i> (SAP_CORE_BC_IAM_RM_DISP_PC)</p> <p><i>Identity and Access Management - Group Management</i> (SAP_CORE_BC_IAM_GRP_PC)</p>
Identity and Access Management	<a href="#">Maintain Deleted Business Users [page 2662]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	<p>SAP_CORE_BC_IAM_UMD</p> <p><i>Identity and Access Management - User Management of Deleted Users</i></p>
Identity and Access Management	<a href="#">IAM Key Figures [page 2703]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	<p>SAP_CORE_BC_IAM_UM</p> <p><i>User Management</i></p> <p>SAP_CORE_BC_IAM_RA</p> <p><i>Role Assignment</i></p> <p>SAP_CORE_BC_IAM_RM</p> <p><i>Role Management</i></p>
Identity and Access Management	<a href="#">Maintain Business User Groups [page 2660]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	<p>SAP_CORE_BC_IAM_GRP_PC</p> <p><i>Identity and Access Management - Group Management</i></p>

<b>Area</b>	<b>Application</b>	<b>Business Role Template ID</b>	<b>Business Catalog ID</b>
Identity and Access Management	<a href="#">Maintain Business Role Groups [page 2664]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_GRP_PC <i>Identity and Access Management - Group Management</i>
Security	<a href="#">Maintain Certificate Trust List [page 2771]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Maintain Protection Allow-lists [page 2763]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Manage Content Security Policy [page 2765]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Maintain Client Certificates [page 2772]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Display Security Audit Log [page 2775]</a>	SAP_BR_ADMINISTRATOR Administrator SAP_BR_DATA_PRIVACY_SPECIALIST Data Privacy Specialist SAP_BR_EXTERNAL_AUDITOR External Auditor	SAP_CORE_BC_SEC_SAL <i>Security Audit Log</i>
Communication Management	<a href="#">Communication Arrangements [page 2595]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Display Communication Scenarios [page 2595]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Communication Systems [page 2598]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Maintain Communication Users [page 2593]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Maintain Extensions on SAP BTP [page 2603]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Display Connectivity Trace [page 2606]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>

Area	Application	Business Role Template ID	Business Catalog ID
Communication Management	<a href="#">Monitor bgRFC Queues [page 2608]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Display Inbound Services [page 2604]</a>	SAP_BR_ADMINISTRATOR Administrator	<i>Communication Management</i> (SAP_CORE_BC_COM)  <i>Communication Management - Display</i> (SAP_CORE_BC_COM_DISP _PC)
i18n Services	<a href="#">Language Configuration [page 2647]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_I18N_LANG <i>Language Configuration</i>
Output Management	<a href="#">Maintain Print Queues [page 2715]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_OM_PRT <i>Output Management - Printing</i>
Output Management	<a href="#">Monitor Email Transmissions [page 2727]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_OM_EQM_PC <i>Output Management - Monitor Email Transmissions</i>
Additional Software	<a href="#">Install Additional Software [page 2560]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAS <i>Additional Software</i>
Custom Code Migration	<a href="#">Custom Code Migration [page 2609]</a>	SAP_BR_IT_PROJECT_MANAGER Project Manager - IT	SAP_CORE_BC_CCM <i>Custom Code Migration</i>
Software Component Lifecycle Management	<a href="#">Manage Software Components [page 2805]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_A4C_BC_MSCL_PC <i>Lifecycle Management - Software Components</i>
Message Monitoring	<a href="#">SOAP Error Log, OData Error Log, and Event Error Log [page 2713]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT Configuration Expert - Business Network Integration	SAP_CA_BC_COM_TECH_ERR_PC  <i>Communication Management - Technical Message Monitoring</i>
Message Monitoring	<a href="#">Message Dashboard [page 2708]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT Configuration Expert - Business Network Integration	SAP_CA_BC_COM_TECH_ERR_PC  <i>Communication Management - Technical Message Monitoring</i>

Area	Application	Business Role Template ID	Business Catalog ID
Message Monitoring	<a href="#">Message Monitoring Overview [page 2709]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT  <i>Configuration Expert - Business Network Integration</i>	SAP_CA_BC_COM_TECH_ERR_PC  <i>Communication Management - Technical Message Monitoring</i>
Message Monitoring	<a href="#">Message Monitoring for Integration Experts [page 2710]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT  <i>Configuration Expert - Business Network Integration</i>	SAP_CA_BC_COM_TECH_ERR_PC  <i>Communication Management - Technical Message Monitoring</i>
Message Monitoring	<a href="#">Assign Recipients to Users [page 2711]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT  <i>Configuration Expert - Business Network Integration</i>	SAP_CA_BC_COM_TECH_ERR_PC  <i>Communication Management - Technical Message Monitoring</i>
Message Monitoring (Emergency Correction)	<a href="#">Message Monitoring (Emergency Correction) [page 2712]</a>	SAP_BR_CONF_EXPERT_BUS_NET_INT  <i>Configuration Expert - Business Network Integration</i>	SAP_CA_BC_COM_EMC_PC  <i>Communication Management - Emergency Message Monitoring</i>
Development	<a href="#">SQL Trace Analysis [page 2622]</a>	SAP_BR_DEVELOPER  Developer  SAP_BR_APPL_SUP_ENG_DE_V_SUP  <i>Application Support Engineer - Development Support</i>	SAP_A4C_BC_DEV_SUP_PC  <i>Development - Analysis and Support</i>
Development	<a href="#">Display Publishing Processes [page 2623]</a>	SAP_BR_DEVELOPER  Developer  SAP_BR_APPL_SUP_ENG_DE_V_SUP  <i>Application Support Engineer - Development Support</i>	SAP_A4C_BC_DEV_SUP_PC  <i>Development - Analysis and Support</i>
Technical Monitoring	<a href="#">Application System Overview [page 2781]</a>	SAP_BR_ADMINISTRATOR  Administrator  SAP_BR_DEVELOPER  Developer	SAP_CORE_BC_TMC  <i>Technical Monitoring</i>

<b>Area</b>	<b>Application</b>	<b>Business Role Template ID</b>	<b>Business Catalog ID</b>
Technical Monitoring	<a href="#">System Workload [page 2784]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">HANA Table Analysis [page 2781]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Capture Request Statistics [page 2785]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Perform System Sizing [page 2786]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">System Outbound Communication [page 2784]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Sampled Work Process Data [page 2782]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i> SAP_BR_DEVELOPER <i>Developer</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Partition HANA Tables [page 2788]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_COT_PC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Schedule Metric Provider Collection [page 2787]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Displaying and Analyzing ABAP Runtime Errors [page 2970]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_TMC <i>Technical Monitoring</i>

Area	Application	Business Role Template ID	Business Catalog ID
Technical Monitoring	<a href="#">HANA Thread Samples [page 2783]</a>	SAP_BR_ADMINISTRATOR	SAP_CORE_BC_TMC
		Administrator	<i>Technical Monitoring</i>
		SAP_BR_DEVELOPER	
Translation	<a href="#">Maintain Translations [page 2793]</a>	SAP_BR_ADMINISTRATOR	SAP_CORE_BC_TRA
		Administrator	<i>Translation</i>
ABAP Dictionary	<a href="#">Manage Database Cache Configuration [page 2552]</a>	SAP_BR_ADMINISTRATOR	SAP_CORE_BC_A4C_DIC
		Administrator	<i>Administration - ABAP Dictionary</i>
ABAP Dictionary	<a href="#">Repair CDS Views [page 2551]</a>	SAP_BR_ADMINISTRATOR	SAP_CORE_BC_A4C_DIC
		Administrator	<i>Administration - ABAP Dictionary</i>

## Related Information

[Business Catalogs for Development Tasks \[page 2548\]](#)

### 6.3.1.1.2 Business Catalogs for Development Tasks

Get an overview of available business catalogs for development tasks and their restrictions.

You assign business catalogs to business roles that are assigned to business users. Business catalogs contain authorizations that define what a business user with a certain business role is allowed to do.

Certain business catalogs are only available in development systems (see [Creating an ABAP System \[page 180\]](#)).

Business Catalogs for Development Tasks

Business Catalog	Authorizations	Restrictions	Availability
<i>Development - ABAP Development Tools</i> SAP_A4C_BC_DEV_PC	ADT Development	No transport request management (transport tasks only)	In development systems only

Business Catalog	Authorizations	Restrictions	Availability
<i>Development - Analysis and Support</i> SAP_A4C_BC_DEV_SUP_PC	Troubleshooting tools such as logs, traces, and the debugger.	-	In all systems
	<p><b> ⓘ Note</b></p> <p>To enable debugging authorization, set the access category Write, Read, Value Help to <i>Unrestricted</i>. Upon creation of the business role, the value is set by default to <i>No Access</i>. See <a href="#">How to Define Authorizations Based on Restrictions [page 2672]</a>.</p>		
<i>Development - API Test</i> SAP_A4C_BC_DEV_TST_PC	Testing ABAP-based APIs released by SAP	-	In development systems only
<i>Development - Class Runner Execution</i> SAP_A4C_BC_DEV_CLA_RU_N_PC	Executing class runners in ABAP Development Tools	-	In all systems
<i>Development - Data Preview - Released Objects</i> SAP_A4C_BC_DEV_DAT_PR_V_PC	Using data preview in ABAP Development Tools	-	In all systems
<i>Development - Development Objects Display</i> SAP_A4C_BC_DEV_OBJ_DIS_PC	Viewing (read-only) development objects in ABAP Development Tools	-	In all systems
<i>Development - Transport Management</i> SAP_A4C_BC_TRN_MNG_PC	ADT Transport Management	No release of transport requests and no customizing requests	In development systems only
<i>Development - Transport Release Management</i> SAP_A4C_BC_TRN_REL_PC	ADT Transport Release Management	No customizing requests	In development systems only
<i>Development - UI Deployment</i> SAP_A4C_BC_DEV_UID_PC	Deployment of UIs into the ABAP system repository	-	In development systems only

Business Catalog	Authorizations	Restrictions	Availability
<i>Development Support - Data Preview - Business Data</i> SAP_A4C_BC_DAT_PRV_DF_T_PC	<p>Usage of the data preview in ABAP Development Tools for objects that are considered to contain business data:</p> <ul style="list-style-type: none"> <li>• CDS views in language version 5</li> <li>• Client-dependent tables in language version 5 with delivery class A and L</li> </ul> <p>Data of CDS views can be viewed with or without application of the DCL.</p>	-	In all systems
<i>Development Support - Data Preview - Cross-Client and Customizing Data</i> SAP_A4C_BC_DAT_PRV_RD_O_PC	<p>Usage of the data preview in ABAP Development Tools for objects that are considered to contain customizing and cross-client data:</p> <ul style="list-style-type: none"> <li>• Client-independent tables in language version 5 with delivery class A and L</li> <li>• Tables in language version 5 with delivery class S, E, and W (metadata)</li> <li>• Tables in language version 5 with delivery class C and G (customizing)</li> </ul>	-	In all systems
<i>Extensibility - Custom Apps and Services</i> SAP_CORE_BC_EXT_TST	<ul style="list-style-type: none"> <li>• Previewing UIs in SAP Business Application Studio and Microsoft Visual Studio Code</li> <li>• Previewing service bindings</li> <li>• Testing custom apps and custom services</li> </ul>	Only services that have their original in the current system	In development systems only

### 6.3.1.1.3 Business Catalog for Key User Tasks

Get an overview of available business role catalogs and their restrictions.

You assign business catalogs to business roles that are assigned to business users. Business catalogs contain authorizations that define what a business user with a certain business role is allowed to do.

Business Catalogs for Key User Tasks

Business Catalog	Authorizations	Restrictions
<i>Extensibility - Custom Apps and Services</i> SAP_CORE_BC_EXT_TST	Testing of custom apps and custom services	Only services that have their original in the current system
<i>Extensibility - Key User Adaptation</i> SAP_CORE_BC_EXT_FLEX	Key user adaptation	

## 6.3.1.2 ABAP Dictionary

[Repair CDS Views \[page 2551\]](#)

[Manage Database Cache Configuration \[page 2552\]](#)

### 6.3.1.2.1 Repair CDS Views

With this app you can view and repair the CDS views that are in inconsistent state.

#### Key Features

You can use this app to:

- **View** the details of the inconsistent CDS views such as the application component, the error category, the person who is responsible, and the package details
- **Search** for a specific view based on the view name or filter the views based on the error category
- **Repair** the inconsistent views

In addition, the app allows you to export the CDS views to spreadsheets.

#### Supported Device Types

- Desktop

#### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-DWB-DIC.

## 6.3.1.2.2 Manage Database Cache Configuration

With this app you can view and manage the database cache configurations.

### Key Features

- a) You can use this app to:
- View the list of caches in the system.
  - Search for a particular cache based on the name or the data source.
  - View the cache-specific information such as total number of records cached and size available for consumption.
  - Create or drop a cache by choosing *Create* or *Drop* respectively.

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-DWB-DIC.

## 6.3.1.3 ABAP Test Cockpit

### [ABAP Test Cockpit Configurator \[page 2553\]](#)

SAP Fiori app for editing ATC configurations.

### [Approve ATC Exemptions \[page 2554\]](#)

### [Manage API Snapshots \[page 2556\]](#)

### 6.3.1.3.1 ABAP Test Cockpit Configurator

SAP Fiori app for editing ATC configurations.

#### Purpose

The *ABAP Test Cockpit Configurator* enables you to maintain ATC configurations using SAP Fiori launchpad.

##### ⓘ Note

The *ATC Configurator* app enables you to maintain multiple configurations. Edits done in transactions `ATC` and `SCI` only affect the **active** configuration.

#### Key Features

1. Assign a check variant.
2. Change priorities.
3. Set priority levels that block or interrupt transport releases.
4. Set a default configuration.
5. Set handling of pseudo comments and pragmas.

#### Access Information

For more information about how to provide access to users and how to implement this app, see [Enable Usage of the ABAP Test Cockpit Configurator App \[page 2554\]](#).

#### Component for Customer Incidents

BC-DWB-TOO-ATF

#### Supported Device Types

- Desktop

## Related Information

[Enable Usage of the ABAP Test Cockpit Configurator App \[page 2554\]](#)

### 6.3.1.3.1.1 Enable Usage of the ABAP Test Cockpit Configurator App

#### Prerequisites

- Access to SAP Fiori launchpad with administrator authorization

#### Overview

1. Create a business role either from a template or from scratch.
  - To use a template, see [How to Create a Business Role from a Template \[page 2668\]](#). Choose the template SAP\_BR\_ADMINISTRATOR\_SW\_DEV.
  - To create a role from scratch, see [How to Create a New Business Role \[page 2667\]](#). Choose ABAP Test Cockpit Configuration (business catalog ID: SAP\_CORE\_BC\_ATC\_CONFIG) as catalog for this role (step 3).
2. Assign the business role you created to a business user.  
(See also: [How to Maintain Business Users \[page 2656\]](#).)

When you log on to SAP Fiori launchpad with the assigned business user, the tile ABAP Test Cockpit Configurator will be available.

### 6.3.1.3.2 Approve ATC Exemptions

With this app you can approve or reject ATC exemptions via the SAP Fiori launchpad.

#### Key Features

With this app you can:

- Approve exemption requests.
- Reject exemption requests.

- See a list of exemptions of all states.
- Apply filters to the list of exemptions, such as exemption state, processor and requester.
- See a justification for an exemption request.
- Add or remove columns to or from the list of exemptions to customize your view.
- See the scope the requester would like their exemption request to be valid for through the object scope (finding, object, sub-object and all objects of a package) (granularity).
- See the scope the requester would like their exemption request to be valid for through the check scope (finding, check message, check) (validity).
- See the system group for which an exemption would be valid for.
- See the expiry date of an exemption. The processor of the request can change the expiry date after request creation.
- Give an assessment for an exemption request.
- Replicate exemptions from an on-premise system to SAP BTP.

#### Note

Use transaction `SATC_EXPORT_XMPT` to download exemptions from an on-premise ATC system. For customers that are on a release between 7.52 and 7.58 with software component SAP\_BASIS, apply SAP Note [3480722](#) first to install this functionality.

## Access Information

For more information on how to enable this app, see [Enable Usage of the Approve ATC Exemptions App \[page 2556\]](#).

## Component for Customer Incidents

If you need support or are experiencing issues, please report an incident under component `BC-DWB-TOO-ATF`.

## Supported Device Types

- Desktop
- Tablet
- Smartphone

## 6.3.1.3.2.1 Enable Usage of the Approve ATC Exemptions App

### Prerequisites

You have access to the SAP Fiori launchpad with a quality manager authorization and you've configured the ATC Developer Scenario. For more information, see [Using an SAP BTP System as ATC Central Check System](#).

### Procedure

1. Create a business role either from a template or from scratch.
  - To use a template, see [How to Create a Business Role from a Template \[page 2668\]](#). Choose the template SAP\_BR\_QUALITY\_MNGR\_SW\_DEV.
  - To create a role from scratch, see [How to Create a New Business Role \[page 2667\]](#). Choose Approve ATC Exemptions (business catalog ID: SAP\_A4C\_BC\_DEV\_XMPT\_APP\_PC) as catalog for this role (step 3).
2. Assign the business role you created to a business user.  
(See also: [How to Maintain Business Users \[page 2656\]](#).)

When you now log on to the SAP Fiori launchpad with the assigned business user, the tile *Approve ATC Exemptions* will be available.

### Related Information

[Approve ATC Exemptions \[page 2554\]](#)

[SAP Note 3480722 - Downport Program to Export ATC Exemptions](#)

## 6.3.1.3.3 Manage API Snapshots

With this app you can create API snapshots of your components. An API snapshot is a snapshot of the external interface of all released objects within a software component. This snapshot should be taken when the released objects within the software component must be kept stable. API snapshots allow you to run compatibility checks for your released objects using the ATC check API\_COMPATIBILITY.

## Key Features

You can use this app to:

- Create API snapshots to be able to use them in compatibility checks
- Regenerate failed APIs
- Set APIs for check relevance to check their content
- Overwrite your API snapshots to recreate already existing snapshots
- Display the changes made to your API snapshots
- Display extracted APIs

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-EXT-REL.

### 6.3.1.3.3.1 Enable Usage of the Manage API Snapshots App

Find out how to enable your *Manage API Snapshots* app in the SAP Fiori launchpad.

## Prerequisites

You already have access to the SAP Fiori launchpad and you have the administrator authorization.

## Overview

1. Create a business role either from a template or from scratch.
  - To use a template, see [How to Create a Business Role from a Template \[page 2668\]](#). Choose the template SAP\_BR\_ADMINISTRATOR\_SW\_DEV.
  - To create a role from scratch, see [How to Create a New Business Role \[page 2667\]](#). Choose API Snapshots (business catalog ID: SAP\_CORE\_BC\_ARS\_PC) as catalog for this role.

- Assign the business role you created to a business user.  
(See also: [How to Maintain Business Users \[page 2656\]](#).)

When you log on to SAP Fiori launchpad with the assigned business user, the tile **Manage API Snapshots** will be available.

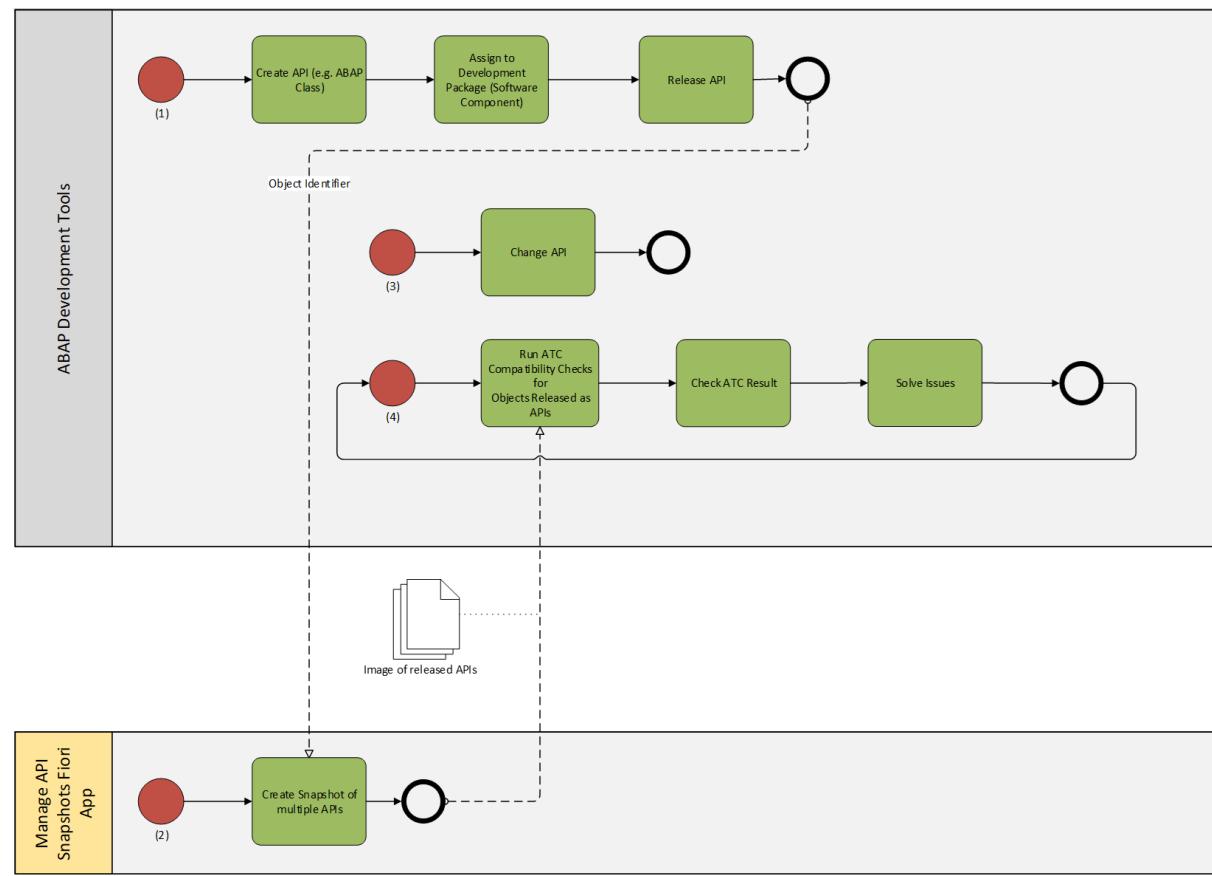
### 6.3.1.3.3.2 Creating API Snapshots

Find out how to create API snapshots using the *Manage API Snapshots* app.

#### Context

To be able to run compatibility checks for your released objects using the ATC check `API_COMPATIBILITY`, you need to create API snapshots of your application components beforehand. This app allows you to generate and manage your API snapshots.

The *Manage API Snapshots* app is part of the business process that helps you to ensure the stability of your released APIs. This overview contains all steps that need to be considered when developing, releasing and reworking APIs for use by others:



## Procedure

1. Open the *Manage API Snapshots* app.
2. Click *Create*. Choose a *Name* for your snapshot and select the snapshot component from the value help.
3. Click *Create* again to create your API snapshot, or select *Create and Generate* to generate a snapshot. The generation process will be shown on the UI. Once the snapshot generation has finished, the action status will change to *Completed*. You can view the snapshot details by selecting the snapshot.

### ⓘ Note

If you release your object and a snapshot doesn't exist already, it's automatically created. This default snapshot appears in your list of snapshots as *SAP Default*. An SAP default snapshot has limited use: you can't regenerate or delete this snapshot, but you can regenerate single APIs of the SAP default snapshot in the *APIs* tab, however. You can set the check relevance of the snapshot as usual. Mind that if at the time of its creation, the SAP default snapshot is the only available snapshot in the respective software component, or if no other snapshot of the respective software component is set to check-relevant, it'll be set to check-relevant automatically.

4. In the *Snapshot API Details*, you can choose to maintain your snapshot in the following way:
  - a. Change the name of your snapshot by clicking *Edit* in the top right corner of your screen in the *General Information* tab.
  - b. Regenerate the extracted APIs of your snapshot in the *Extracted APIs* tab. To regenerate your APIs, select the ones in question and click *Regenerate*. Remember that the regeneration of APIs is irreversible.
  - c. Create a note for other people to view by selecting the *Notes* tab and posting a note.
  - d. View the change history of your snapshot by selecting the *Change History* tab.
  - e. View your error logs in the *Generation Log* tab.
  - f. Regenerate your API snapshot by selecting *Regenerate* in the top right corner of your screen.
  - g. Set your API snapshot to check or uncheck the relevance by selecting *Set Check Relevance*.

### ⓘ Note

Only one snapshot per software component can be set as check-relevant.

- h. Delete your snapshot by selecting *Delete*.

You've now created a snapshot that you can use for your compatibility check.

### 6.3.1.4 Additional Software

[Install Additional Software \[page 2560\]](#)

### 6.3.1.4.1 Install Additional Software

With this app you can only display the list of the apps that are available for download. This helps you to better integrate your apps with other programs you need for your daily business.

#### Key Features

You can use this app to:

- Download the current version of the SAP Cloud Print Manager

→ Tip

For more information, see the *SAP Cloud Print Manager Quick Guide* that you can call up directly in the application by clicking *Help*. This document is only available if you have installed SAP Cloud Print Manager.

- Download the current version of ABAP Development Tools

#### Supported Device Types

- Desktop

#### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CCM-PRN-OM.

### 6.3.1.5 Application Jobs

Find out more about application jobs in SAP and how they can be helpful.

Many business activities have to be run periodically. To avoid starting these activities manually each time, you can create application job templates and schedule them as application jobs in the *Application Jobs* app.

An application job template can be scheduled as an application job. In the scheduling dialog, you as a business user can determine a periodic start condition (such as every first Monday of the month at 1:00), and you can set appropriate parameter values for the application job execution. The application job will then be run automatically without further manual interaction, whenever the start condition is fulfilled.

SAP also offers the *Application Job Templates* app. In this app, you can create new templates by saving an existing template with new parameter values. You can also connect existing templates to multistep templates. Running a multistep template means running all steps sequentially.

[Application Jobs \[page 2561\]](#)

[Application Job Templates \[page 2566\]](#)

[Maintain Job Users \[page 2569\]](#)

### 6.3.1.5.1 Application Jobs

You can use this app to monitor all application jobs from all specialized application jobs apps (example: Schedule Overhead Accounting Jobs,...) that business users have scheduled. You can also cancel application jobs.

#### Key Features

##### ⓘ Note

You can use the scheduling, copying and restarting features in the *Application Jobs* app only if you have been granted the necessary authorization via a business role. For more information on how to get the necessary authorizations for this app, see [Maintaining Authorizations \[page 2565\]](#).

You can use this app to:

- Schedule jobs based on a job template: Select a job template, define the description, start date, start time and recurrence of the job, and enter job-specific selection criteria and parameters
- Work with personalized job templates: Create and personalize a job template, save personalized job template for later use, and share personalized template

##### ⓘ Note

The parameters under *Scheduling Information* can't be saved as a part of a template.

- Schedule jobs with a custom factory calendar: In the *Application Jobs* app, define the scheduling information, change an existing calendar, create a new one, and define execution days
- Arrange job series that take different time zones into account
- Monitor jobs
- Display logs and results: You can navigate to application job logs and application job results by clicking on the icon in the *Log* or the *Results* column. The icon can be either an error icon, a success icon, or an info icon. Selecting the icon, you'll navigate to the specific logs or the specific results of the selected job.
- Copy a job
- Cancel a job
- Restart Job Chains

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-APS-APJ.

### **6.3.1.5.1.1 Creating an Application Job**

Find out how to create your own application jobs in the *Application Jobs* app.

#### **Context**

You can use the *Application Jobs* app to schedule and monitor application-related jobs. If you have manual activities that you often need to do at a specific time, the app can reduce your workload by running these tasks smoothly in the background. You can plan regular jobs which allows you to concentrate on other tasks.

#### **Procedure**

1. Open the *Application Jobs* app.
2. Choose *Create* to create a new application job.
3. Select a job template and enter a job name.
4. Specify your preferred scheduling options. You can define the recurrence of your job by selecting *Define Recurrence Pattern*.
5. In the next step, you can adjust the parameters of your application job.
6. You can choose to let the system check if your entries are consistent. If possible, the system fills in default values for the parameters. To choose this option, click *Check*.
7. You can choose to save the application job as a template, or manage your templates. To do so, select *Template*.
8. Finally, to schedule your application job, select *Schedule*.

## 6.3.1.5.1.2 Job Cancellation

### Job cancellation details

A job can be canceled either by the system or by a user. If a job is canceled by the system, it gets the status *Failed*. If you want to cancel a job, please keep in mind that the system behavior depends on what exactly you want to cancel:

- If you cancel a single job with the status *Scheduled*, this job will be canceled and removed from the job list.
- If you cancel a single job with the status *In Process*, this job will be canceled, will get the status *Canceled* and will remain in the job list.
- If you cancel a job that has the status *Scheduled* and is a part of a job series, all jobs with the status *Scheduled* will be canceled and removed from the job list. All jobs with other statuses will remain in the list and won't be canceled.
- If you cancel a job that has the status *In Process* and is a part of a job series, a job with the status *In Process* will be canceled and will get the status *Canceled*. All other jobs of the series with other statuses will remain in the list.

To cancel a job, please select the job to be removed and choose *Cancel* from the main toolbar.

#### ⓘ Note

Please note that the jobs with the status *Finished* cannot be canceled because they have already been run.

## Related Information

[Application Jobs \[page 2561\]](#)

## 6.3.1.5.1.3 Regular Job Deletion

Get an overview of the regular job deletion

All jobs that are not removed by a user remain in the system until a technical cleaning job runs. The time when this job is run by the system depends on the periodic granularity of the jobs that should be deleted. Currently, the following timeframes apply:

### Regular Job Deletion

Periodic Granularity of the Job to Be Deleted	Job is Automatically Deleted After
Not periodic	14 days
Minutes	7 days
Hours	14 days

Periodic Granularity of the Job to Be Deleted	Job is Automatically Deleted After
Days	Periodic value*24 days
Weeks	Periodic value*7*24 days
Months	Periodic value*31*24 days
Every X week in every Y month	Periodic value*31*24 days

## Example

You have scheduled a job that runs every two weeks and ends after 20 runs. The first run starts in the calendar week 2 and ends in the calendar week 40. Each job run stays in the system for the certain retention time. Retention period is always set to 24 runs. Retention time=periodic value \* retention periods \* weekdays [days] =  $2 * 24 * 7$  [days] = 336 [days] = 48 [weeks]. The first job run will be deleted a day after 48 weeks since this job run took place.

### ⓘ Note

This is the default behaviour. If the application chooses to have different schemes, this is documented in the application-specific documentation.

## 6.3.1.5.1.4 Restart Job Chains

Find out how to restart job chains.

When an application job or a chain of application jobs consisting of one or more steps has the status *Failed*, *Canceled* or *User Error*, you can restart it. To do so, select the job and choose one of the following options from the drop-down menu:

- From Error Step: All steps before the error step will be skipped and all other steps (including the error step) will be executed.
- After Error Step: All steps before the error step including the error step will be skipped and all other steps after the error step will be executed. This option is not available for single step jobs.

The new job will be executed immediately with the original job user.

### ⓘ Note

You can't change the starting or application parameters.

## Related Information

[Application Jobs \[page 2561\]](#)

### 6.3.1.5.1.5 Maintaining Authorizations

Find out which authorizations you need to maintain in order to use the [Application Jobs](#) app.

## Context

To change access to the objects of other users used in the [Application Jobs](#) app, create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_JCE to it. You can modify the provided restriction types according to your needs.

## Procedure

1. Create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_JCE to it. The restriction type Job Catalog Entry/Application Job Part contains the two fields Application Job Catalog Entry and Application Job Part. For more information, see [Maintain Business Roles \[page 2666\]](#).

### ⓘ Note

With this restriction type, you'll be able to provide the necessary authorizations for certain actions on the application jobs of other users that are based on a specified job catalog entry. You can display the result lists of these application jobs.

Alternatively, you can also create a display only-role using this restriction type.

2. Next, define your business role with the restriction fields Application Job Catalog Entry and Application Job Part mentioned above. To do so, select [Maintain Restrictions](#). In the Application Job Catalog Entry field, you can choose the job catalog entries, and in the Application Job Part field, you can choose for which job part access should be granted. You can choose the values JOB (Job Details), SPOOL (Result List), and APPLOG (Log) by clicking the [Edit](#) icon next to the fields.
3. If you want to have different business users for the job owner and the job user, the job owner needs the authorization to schedule the application job for the job user. In this case, the job owner needs the restriction type [Create Application Jobs for Other Users](#).
4. Go back and select [Save](#). Now, you've maintained your business role, and you can assign the role to a business user. For more information on authorizations, see [3028505](#).

### ⓘ Note

The private templates of other users aren't shown in the [Application Jobs](#) app, even if the authorization to view these private templates was granted via SAP\_CORE\_BC\_APJ\_TPL. You can view the private

templates of other users only in the *Application Job Templates* app. For more information, see [Application Job Templates \[page 2566\]](#).

## Related Information

[Maintain Business Users \[page 2655\]](#)

[Maintain Business Roles \[page 2666\]](#)

### 6.3.1.5.2 Application Job Templates

You can use this app to create your own application job templates. The application job templates consist of a set of parameters that you can set for the assigned job catalog entry. This app is mandatory to create multi-step templates which can then be submitted via the *Application Jobs* app. A multi-step template consists of several other templates that run in a series, one after the other. Each step could be a single-step template, or another multi-step template that was created with the *Application Job Templates* app before.

## Key Features

You can use this app to:

- Create and customize a job template
- Create multi-step templates
- Change the template parameters
- Transport job templates, if supported: For more information, see [2999966](#)
- Display a list of templates you have read access to
- Display the layer code of the template:
  - *Global*: a template with this layer code is predelivered by SAP and therefore can't be edited or deleted
  - *Private*: a template with this layer code is visible for all users within the *Application Job Templates* and the *Application Jobs* apps, but only the author can edit or delete the template

#### ① Note

With the new restriction type, you can define a full authorization for job templates, such that a system administrator can display, change or delete private templates of other users

- *Shared*: a template with this layer code is visible to all users that have the required authorizations to use the *Application Job Templates* and the *Application Jobs* apps
- *Reference*: a template with this layer code is visible for all users within the *Application Job Templates* and the *Application Jobs* apps, but only the author can edit or delete the template. It can't be scheduled in the *Application Jobs* app. It can be used as a copy pattern to create other templates.
- Delete a job template

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-APJ.

## Related Information

[Creating an Application Job Template \[page 2567\]](#)

### 6.3.1.5.2.1 Creating an Application Job Template

Find out how to create your own application job templates in the *Application Job Templates* app.

## Context

You can use the *Application Job Templates* app to create a job template for a specific business purpose so that you can schedule concrete jobs by reusing the job template.

## Procedure

1. Open the *Application Job Templates* app.
2. Choose *Create* to create a new job template.
3. Enter a name and specify your preferred scheduling options.
4. Tick the *Shared Template* checkbox if you want other users to be able to view or edit the job template. If you share the template, all users who have access to the *Application Job Templates* app, and who have the appropriate authorizations, can make the same changes as you.
5. Select *Maintain Steps* and choose *Add* to add an existing template as step. This step can be repeated many times to create a multi-step template.
6. Click *Save Job Template*. Now, you can view each parameter of your selected templates in the *Parameter Section* and edit the values according to your needs.
7. If you are done editing, you can now choose whether to check your parameters, or display, save, or cancel your job template.

#### ⓘ Note

If you want to schedule multistep templates with the [Application Jobs](#) app, you'll need the appropriate authorizations for all steps.

## 6.3.1.5.2.2 Maintaining Authorizations

Find out which authorizations you need to maintain in order to use the [Application Job Templates](#) app.

### Context

In order to grant access and authorizations to the [Application Job Templates](#) app, create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_TPL to it.

### Procedure

1. Create a new business role assigning the business catalog SAP\_CORE\_BC\_APJ\_TPL to it. For more information, see [Maintain Business Roles](#) linked below.
2. Next, select [Maintain Restrictions](#) to provide [Write](#), [Read](#), or [No Access](#) to templates using the restriction type APPLICATION JOB TEMPLATE LAYER.
3. Go back to [Save](#) your template.

#### ⓘ Note

You will always be able to see your own templates and the templates delivered by your software provider without having to maintain further authorizations.

#### ⓘ Note

The possible values for the layer field are [Global](#), [Shared](#), and [Private](#). These are described in [Application Job Templates](#) linked below. The restriction type [Global](#) is predelivered by SAP and therefore can't be edited or deleted, but you can modify the restriction types [Shared](#) and [Private](#) to your needs.

#### ⓘ Note

If you want to schedule multistep templates with the [Application Jobs](#) app, you'll need the appropriate authorizations for all steps.

## Related Information

[Maintain Business Users \[page 2655\]](#)

[Maintain Business Roles \[page 2666\]](#)

[Application Job Templates \[page 2566\]](#)

### 6.3.1.5.3 Maintain Job Users

With this app, you can change the owner and the user of application jobs that you've created from a job template. This can be helpful if, for example, a person changes the department within a company or leaves the company, and the corresponding business user has to be deleted.

#### Key Features

You can use this app to:

- Change either the job owner or the job user of an application job, or change both
- Delete an application job

#### Supported Device Types

- Desktop
- Tablet

#### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-APJ.

### 6.3.1.5.3.1 Editing Job Users

Find out how to change the user of your application job, or delete your job.

#### Prerequisite

Before changing the user of a job, ensure that the target user has the correct authorizations to run the job. Otherwise, the job will fail.

To be able to see the app, make sure that the target user is assigned to a business role that has the business catalog SAP\_CORE\_BC\_APJ\_USR\_PC included.

#### Procedure

1. Open the [Maintain Job Users](#) app.
  2. Select the scheduled jobs that you want to work with, and choose one of the following actions:
    - [Change Job Owner](#) to change the owner of your application job
    - [Change Job User](#) to change the business user of your application job
    - [Change Job Owner and User](#) to change both the owner and the user of your application job
    - [Delete](#) to delete your application job
- When clicking on an application job, you'll be navigated to the [Details](#) page of the [Application Jobs](#) app.
3. If you've decided to change the owner or the user of your application job, or if you want to change both, a dialog opens. Enter the new job owner and user by selecting them from the value help. Mind that in case of a job user change, you can only select users that have the appropriate start authorization for a job. Job owners don't need this authorization, but they need the restriction type `Create Application Jobs for Other Users` instead. For more information, see **Maintaining Authorizations** of the [Application Jobs](#) documentation. App-specific authorizations are not checked. You can choose to only display these users by filtering for [Only Valid](#).

##### Note

If a job owner is a communication user, the owner can't be changed, since the connection to the corresponding communication arrangement of the type SAP\_COM\_0064 would be broken.

4. You can change multiple application job owners and users at the same time. To do this, select all users you want to change and choose [Change Job Owner and User](#). If the job owner differs from the job user, the job owner must have the appropriate authorization given by the restriction type `Create Application Jobs for Other Users` of business catalog SAP\_CORE\_BC\_APJ\_JCE.
5. You can choose to display the change history of your application job by selecting [Change History](#). All changes that were made to the owner and user in the past are shown in the change history. The change history shows the user type, and the new and old user IDs. You can view who changed the user, and when it was changed. The type of change is shown, too.
6. You can delete multiple application jobs at the same time. To do this, select all jobs you want to delete and choose [Delete](#).

You've now maintained your job user.

## Related Information

[Application Jobs \[page 2561\]](#)

[Maintaining Authorizations \[page 2565\]](#)

### 6.3.1.6 Business Configuration

Find out which lifecycle management tasks a business process configuration expert (business role SAP\_BPC\_EXPERT) needs to perform for the administration of business configurations.

#### Context

Business configuration is the customization of business applications. In enterprise software, business configuration refers to a predefined set of configuration options that affect its functionality and behavior. You can maintain custom business configuration content in your SAP BTP ABAP Environment to enrich your extensibility scenarios. This includes both functionalities provided by SAP and custom business configuration objects developed using ABAP Development Tools (ADT).

If you're looking for more information on how to implement custom business configuration objects, please refer to [Custom Business Configurations App \[page 2574\]](#).

#### Requirements

The administration of your business configuration has the following requirements:

- The business configuration is client-dependent. After import, it's only available in the client where the import took place. In contrast, development objects are not client-dependent.
- Business configuration content is usually not directly maintained in a productive system, but created in a development system, and then transported to the test system and productive system. For more information on how to set up your system landscape, please refer to the scenarios described in [Setting Up and Working with Your Landscape](#).
- Changes to business configuration content are recorded onto transport requests of the type Customizing. They can be released in the source system and imported in any target system. For more information on how to set up a suitable software component for transport, please refer to [Software Components](#).

## Procedure

You as a business process configuration expert are responsible for the lifecycle management of business configuration content. First, you need to create transport requests of the type **Customizing** that contain corresponding transport tasks. Once a suitable transport task has been assigned, you can proceed with the creation and adjustment of content. When the content is ready, the transport tasks and the transport requests can be released. The changes will be written to the remote repository corresponding to the software component that was used for transport. The content can then be imported into target systems simply by pulling the changes for the software component that was used. This procedure is valid regardless of the specific landscape setup or whether software components of the type **Development** or **Business Configuration** are used.

### ⓘ Note

It's possible to maintain local business configuration changes which can't be transported to other systems. If no software component of the type **Business Configuration** is cloned in the source system, you can still create a customizing transport request in the [Export Customizing Transports](#) app. This request will have no target. It can be used to record business configuration changes, but its release will have no effect.

The following sections will introduce the applications needed for both the maintenance and lifecycle management of business configuration content.

### 6.3.1.6.1 Business Configuration Change Logs

With this app, business process configuration experts can keep track of content changes in your business configuration tables. Table data change logs can be used to check when data was changed, which kind of data was changed, or by whom. This is made visible in a table.

#### Access Information

To access the app, you need to have the following business catalog assigned to your user:

SAP\_CORE\_BC\_BCT\_LOG\_PC - Business Configuration - Change Logs.

This business catalog is available in the business role template: SAP\_BR\_BPC\_EXPERT.

To display the change logs for tables, users must have the authorization for the object S\_TABU\_NAM with **Display Change Documents** for ACTVT and the name of the table for TABLE.

#### Key Features

You can use this app to:

- Display which changes were made to the records of your business configuration tables.

- Display when changes were made to the records of your business configuration tables.
- Display who made changes to the records of your business configurations tables.

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CUS-TOL-ALO.

### 6.3.1.6.1.1 Working in the Business Configuration Change Logs App

#### Prerequisites

The *Log Changes* checkbox has to be enabled in the technical settings for the table:

- For tables delivered by SAP: SAP defines whether change logging is enabled.
- For tables you have created: You need to enable change logging.

#### ⓘ Note

*Log Changes* checkbox can be enabled or disabled in the [Technical Table Settings](#) ADT editor by expanding the entry of your table in the project explorer.

#### Displaying a Change Log

Find out how to display the change log of your business configuration tables.

#### Procedure

1. Open the [Business Configuration Change Logs](#) app.
2. Use the search bar to find a configuration table. You can search based on the table name or a specified date range.
3. Select a configuration table to see its change log details.
4. You can filter the change log details based on criteria such as field name, changed date, changed by, and more.

## Download Change Logs

Find out how to display the change log of your business configuration tables.

### Procedure

1. Open the *Business Configuration Change Logs* app.
2. Select the button *Download Change Logs*. Once selected you will be navigated to the *Business Configuration Change Logs Overview* app.
3. Select *Create* to schedule a job.
4. Select a *Job Template* and a *Job Name*, then select *Step 2*.
5. You can either start the job immediately by selecting it or select a specific start date and time. Select *Step 3* to continue
6. In the *Parameters* section select a start date and time and an end date and time. All the change logs given in this time window will be downloaded.
7. To download the change logs for a specific user, provide their *Username*. When no username is provided, all change logs from every user will be downloaded
8. The *Only Actual Changes* option allows to download logs that contain only the records that have been changed e.g. inserted, updated, deleted etc. This option excludes logs for unchanged records
9. Once all the Parameters are entered, select *Schedule* to start the job immediately or at the provided time given at *Step 4*.

### Result

In the Business *Configuration Change Logs Overview* app, you can see the scheduled jobs. Once the job is finished you can preview the change logs or download the change logs.

If you need support or experience issues, please report an incident under component BC-CUS-TOL-ALO.

## 6.3.1.6.2 Custom Business Configurations App

Find out how to use the *Custom Business Configurations* app.

### Purpose

The *Custom Business Configurations* app serves as an entry point to the *Business Configuration Maintenance Object* [page 866] provided by the custom applications or partners. You can use the app to adjust these configuration objects to change and influence the system behavior. For more information, see also *Creating Business Configuration Apps with ABAP RESTful Application Programming Model* and *Custom Business Configurations App* [page 864].

## Access Information

The following business catalog needs to be assigned to your user to access the app:

SAP\_CORE\_BC\_BCT\_MBC\_PC.

The business catalog is contained in the business role template: SAP\_BR\_BPC\_EXPERT.

For a business configuration maintenance object to be listed in the *Custom Business Configurations* app, you require the necessary authorizations regarding the service of the business configuration: [Business Configuration Maintenance Object \[page 866\]](#) > [Provide Authorizations for a Business Configuration](#).

## Procedure

1. Open the *Custom Business Configurations* app.
2. You'll see a list of all business configurations that can be adapted.
3. Use the *Search* bar to find a certain business configuration or select the one you would like to work on from the list.

### ⓘ Note

Depending on your respective role and access rights, you may only be able to edit certain business configurations.

4. You can now adjust the business configuration you have selected. Simply add, update, or delete entries.

## Intent Navigation

You can use the parameter `TechnicalIdentifier` for the semantic object `BusinessConfiguration` with the action `maintain`. Intent navigation can be used to directly navigate to the maintenance view of the business configuration specified with the parameter `TechnicalIdentifier`. The attribute `TechnicalIdentifier` is available as a column in the list report of the *Custom Business Configurations* app.

## Display Change Logs

For a selected business configuration, the action *Display Change Logs* is available in the header section if

- for at least one database table of the RAP BO of the Business Configuration Maintenance Object Table Logging is active
- the *Business Configuration Change Logs* app is configured as a valid navigation target for the user for the given device

With this action, you can navigate to the *Business Configuration Change Logs* app.

## Show Documentation

For a selected business configuration, the action [Show Documentation](#) is available in the header section if a knowledge transfer document exists for the business configuration maintenance object. With this action the content of the knowledge transfer document is displayed. For more information, see [Documentation of Business Configuration Maintenance Objects](#).

To provide context-sensitive in-app help you can use [SAP Companion](#).

## Grouping

In the table view settings dialog you can group the business configuration maintenance objects by their attribute `Configuration Group`. This allows the list to be structured in a simple hierarchical manner. If this attribute is not maintained for a business configuration maintenance object, it will be part of a generic group with the header [Not assigned](#).

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CUS-TOL-MBC. If you experience issues updating a specific business configuration, please use the component that is displayed in the error message.

### 6.3.1.6.3 Manage Number Range Intervals

You can use this app to get an overview of number range objects and maintain number range intervals.

## Key Features

You can use this app to:

- Display properties of number range objects
- Display, create, change and delete number range intervals
- Change the number range status of a number range interval
- Display the change history

## Supported Device Type

- Desktop

## Prerequisites

To access the [Manage Number Range Intervals](#) app, assign the SAP\_CA\_BC\_IC\_LND\_NUM\_PC (Number Range Management – Configuration) business catalog to a proper business role and - if required - restrict it to specific number range objects.

To maintain number range intervals and to create transport requests, assign business role BR\_BPC\_EXPERT.

For more information, see:

- [Maintaining Number Range Intervals \[page 2578\]](#)
- [Changing the Number Range Status \[page 2579\]](#)
- [Checking Properties of a Number Range Object \[page 2577\]](#)

### 6.3.1.6.3.1 Checking Properties of a Number Range Object

To check the properties of a number range object, proceed as follows:

1. Start the [Manage Number Range Intervals](#) app from the [Business Configuration](#) app group on the Fiori launchpad.
2. Choose a number range object from the list and click the arrow-button on the right-hand side of the entry. The [Properties](#) section is displayed.

#### ⓘ Note

You can only display properties. You don't have the option of changing them.

The following parameters are displayed:

- Subobject exists: The number range object distinguishes between further subobjects.
- Year-Dependent: This parameter is set to [Yes](#) if the number range intervals are distinguished according to to-fiscals.
- Subobject as Prefix: This parameter is set to [Yes](#) if determined numbers consist of the prefix (name of subobject) and the numbers.
- Number of Digits: This parameter determines the length of an interval.
- Rolling: This parameter is set to [No](#) to prevent the number range object intervals from starting from the upper limit automatically.
- Buffering Type: This parameter shows whether main memory, parallel, or no buffering is supported.
- Number: In case of parallel and main memory, this parameter determines how many numbers are reserved in the buffer for the intervals.
- Warning at Remaining: This parameter displays the percentage of numbers remaining in a number range. When reaching the percentage in number assignment, a warning is given. Assuming an interval

from 1 to 1000 and a percentage of 10 (%) is used, a notification will be triggered if number 900 has been reached.

You can maintain number range objects using the ABAP Development Tool (ADT) or APIs. For more information, see

- ADT:  
[Working with Number Range Objects](#)
- API:  
[Maintaining Number Range Objects](#)

### 6.3.1.6.3.2 Maintaining Number Range Intervals

You can structure the defined character set of a number range object in intervals. For more information on intervals, see [Intervals](#).

To maintain a number range interval, proceed as follows:

1. Start the [Manage Number Range Intervals](#) app from the [Business Configuration](#) app group on the Fiori launchpad.
2. Choose a number range object from the list and click the arrow-button on the right-hand side of the entry. The [Properties](#) section is displayed.
3. In the [Number Ranges](#) section, press [Create](#) to create a new interval. Assign a two-character ID as interval number and a lower and upper limit. If the current number range object supports subtypes or is year-dependent, assign a subtype and a year accordingly.

#### ⓘ Note

Don't set the [External Interval](#) flag if you want to use an internal number assignment. That means that during number determination within an application program using that interval, the number range framework automatically assigns a number from the associated number range object to the data record created by the user. Only digits are assigned from internal intervals of the number range object.

For an external number assignment, an application selects a number for a new data record by their own. This can be useful if you want more information to be included in the structure of the number. The number range framework then checks whether the selected number is in an interval of the number range object that is defined as external. However, it does not check whether this number has already been used. Digits and also letters and special characters are assigned from external intervals of a number range object.

4. Press [Save](#) to save the new interval.

If you want to delete an interval mark it by using the radio button and press [Delete](#).

If you want to change an interval, open it by clicking the arrow-button on the right-hand side of the entry. Then press [Edit](#) if you want to change the limits or the external flag.

### 6.3.1.6.3.3 Changing the Number Range Status

The *Number Range Status* or *Number Range Level* is the last number used by an interval.

#### ⓘ Note

Only change this number in case of inconsistencies.

To change the number range status, proceed as follows:

1. Start the *Manage Number Range Intervals* app from the *Business Configuration* app group on the Fiori launchpad.
2. Choose a number range object from the list and click the arrow-button on the right-hand side of the entry. The *Properties* section is displayed.
3. Choose a number range interval from the list and click the arrow-button on the right-hand side of the entry.
4. Click the *Change Number Range Status* button. In the pop-up, enter a new interval level and press *Change Number Range Status*.

### 6.3.1.6.4 Upload Business Configuration

#### Context

You want to maintain Business Configuration data in the cloud system. However, you do not want to manually maintain it, for instance, due to data volume. You may have already maintained your own configuration tables in the backend of your ERP system and now simply want to import the configuration content from a file into the corresponding tables in your system.

#### Purpose

The **Upload Business Configuration** app offers an easy way to import the configuration content via ".xlsx" file upload. Such a file may have been generated from a leading ERP system.

#### Access Information

The following business catalogue needs to be assigned to your user to access and use the app:

- SAP\_CA\_BC\_IC\_LND\_PC - Business Configuration – Customizing

In addition, you need an open customizing transport, which can be created by a user who has the following business catalogues assigned to them:

- SAP\_CORE\_BC\_BCT\_TRN\_MNG\_PC (Business Configuration - Transport Management) – Assigned to new role SAP\_BCR\_CORE\_BCT\_TRN\_MNG\_PC
- 2. SAP\_CORE\_BC\_BCT\_TRN\_REL\_PC (Business Configuration - Transport Release Management) – Assigned to new role SAP\_BCR\_CORE\_BCT\_TRN\_REL\_PC

All of these business catalogs are contained in the business role template SAP\_BR\_BPC\_EXPERT.

## Prerequisites

**Use case 1:** Export content from an SAP system on-premise, e.g. SAP S/4HANA, and import it using this app.

- The file to be uploaded via the [Upload Business Configuration](#) app is available in an external format. Here's how you can generate such a file from SAP systems based on the ABAP platform:
  - The configuration content of the maintenance object is exported via transaction [SM30 > Table View > Export \(Spreadsheet\)](#) in ".xlsx" file format. If the [Export \(Spreadsheet\)](#) menu is not available, you can export the configuration content using [Print](#). ([SM30 > Table View > Print > Local File > Save List in File](#).
  - Text tables can be exported via transaction [SE16 > Spreadsheet](#). Keep in mind that the [client \(MANDT\)](#) field needs to be removed from the exported spreadsheet.

**Use case 2:** Import content for your custom C tables using this app.

- Create a spreadsheet with table fields as column headers, excluding the field of type "client".
- To upload the content to objects which are in a customer namespace, you need to have the authority for authorization object S\_TABU\_NAM. TABLE = <Table Name> ACTVT = '02'. Please contact your administrator or IAM expert.

### ⓘ Note

The [Upload Business Configuration](#) app only accepts files with the extension ".xlsx". The uploaded file can contain a maximum of 5000 rows. If there are more rows in the exported file, it needs to be split as per the 5000 row limitation. Apart from explicitly enabled SAP namespace objects, the app only allows uploads for customer namespace tables that have delivery class "C", i.e. "customizing".

### ⓘ Note

Business Configurations for tables with field types STRING or XSTRING are not supported and therefore can't be uploaded.

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CUS-TOL-BCD.

## 6.3.1.6.4.1 How to Upload Business Configurations

You want to import your configuration content via ".xlsx" file upload.

1. Open the [Upload Business Configuration](#) app.
  2. In the [Select Object](#) step, open the value help to select the [Object Name](#) from the allowed list of objects.
  3. Select one of the following operations from the drop-down menu:
    - [Upsert \(Update or/and Insert\)](#): Updates existing entries and inserts new entries.
    - [Insert \(Insert Only\)](#): Inserts new entries but doesn't overwrite/update existing entries.
    - [Replace \(Replace Existing Records\)](#): Replaces any existing entries with the new list of entries. Keep in mind that existing entries will be deleted. This action cannot be undone.
- Click [Step 2](#) to continue.
4. Select [Download File Template](#) to download the file with a templated example of the object selected. See the [Download File Template](#) section below for details.
  5. Upload the adjusted downloaded file which contains the configuration content for the selected object. Once you uploaded it, select [Step 3](#).
  6. In the [Select Configuration Data](#) step, the configuration content from the file you uploaded is displayed. You can unselect the rows you don't want upload and click on [Review](#) to continue.
  7. Review the information. If everything is correct, select [Deploy](#).
  8. Based on the system and object settings, you may get a pop-up with a defaulted transport request or no pop-up appears. You can change the transport request by selecting another one from the list. For more informations see [Transport Request](#) section below.
  9. Select your transport request and confirm it with the OK button. The changes will be sent for deployment to be captured in the provided transport request.

## Transport Request

The transfer request pop-up is based on the system and object settings. If a transport request is mandatory for the selected object and the transport is not hidden, then the transport request which is selected by default is either the [Open Default Customizing/Cross-Client Customizing Request](#) in the [Export Customizing Transports](#) app or the task which was last used by the current user. If no pop-up appears, the default request was selected and you'll get a confirmation message.

In case there is no available transport request in the transport dialog, or you want to create a new request, select [Create](#) in the transport dialog. You'll then be redirected to the [Export Customizing Transports](#) app in a new tab. Once redirected, you can then create a new request.

After creating a new transport request in the [Export Customizing Transports](#) app, you can find it in the [Upload Business Configuration](#) app by using the value help.

## Download File Template

By selecting [Download File Template](#) a pop-up will appear.

## Choose Required File Template Option

 Files downloaded with all fields can't be used for upload

Download with

Fields allowed for upload

All Fields

Data

[Continue](#) [Cancel](#)

There are two options to download the file

1. *Fields allowed for upload* downloads the file which can be used for uploading configuration content. You can always adjust the content according to your requirements before uploading it.
2. *All Fields* provides the additional capability to download the file with all the fields of the selected object, including read-only field

If the *Data* option is selected the file will be downloaded with the current configuration. If this option is deselected a blank file will be downloaded.

### 6.3.1.6.5 Export Customizing Transports

With this app, business process configuration experts can manage business configuration changes recorded in requests.

Business configuration changes are recorded in requests, depending on the category to which the customizing objects belong. Unlike ABAP repository objects, customizing objects recorded in a request are accessible to other business users.

### Access Information

The following business catalogs are available to manage the customizing requests:

- Display/Edit/Delete: SAP\_CORE\_BC\_BCT\_TRN\_MNG\_PC - Business Configuration - Transport Management
- Display/Edit/Delete/Release: SAP\_CORE\_BC\_BCT\_TRN\_REL\_PC - Business Configuration - Transport Release Management

These business catalogs are contained in the business role template: SAP\_BR\_BPC\_EXPERT.

## Key Features

You can use this app to:

- Display a list of customizing requests
- Create new customizing transport requests
- Display the customizing tasks, objects, attributes, notes and logs of a customizing transport request
- Display the objects recorded in a customizing task
- Display the table keys recorded for an object
- Add notes to a customizing request
- Check customizing transport requests
- Release customizing transport requests
- Assign a transport request to your user
- Create tasks for other users
- Change the transport category
- Take over tasks and transports from other users and assign them to your user
- Check the consistency of all open requests

## Default Customizing Request

Since configuration changes recorded in requests of the type `Customizing Request` are not lockable, it might happen that several business users record their configuration changes in different requests.

Recording the configuration changes in different requests might create dependencies between these requests, however to avoid such dependencies, a new transport category is introduced.

Any new request created from this app is categorized as *Default*: In a client, there can be only one open default request of the type `Customizing Request`.

Business users can record their configuration changes to this default request to avoid any dependencies.

If you need to perform an emergency fix, record the configuration changes in a request which is not categorized as *Default*. Default requests are listed in the column *Transport Category*.

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CUS-TOL-CTO.

## 6.3.1.6.5.1 Working in the Export Customizing Transports App

Find out how to create, release, or merge customizing requests using the *Export Customizing Transports* app.

- If a software component of the type **Business Configuration** is available in the system, the created request will target this software component. If no such software component is available, the request will be created without any target.

### Changing Client-Specific Business Configurations

The client settings determine whether changes to business configurations are allowed or recorded automatically in a request.

If the automatic recording for the client is activated, you have to select a request number for the recording whenever you save changed client-specific configurations. If there is no request number selected, the configurations can't be saved.

If the automatic recording for the client is not allowed, you can save changed client-specific configurations without selecting a request number.

### Display Request

1. Navigate to a request's object page to display its tasks, objects, attributes, notes and logs.
2. Select a customizing task to display the objects recorded in it.
3. Select an object to display the table keys recorded in it.

### Create Request

1. Open the *Export Customizing Transports* app. A list of customizing requests is displayed, with the latest request shown first at the top.
2. Use the search bar to filter for specific requests based on search criteria like the *Request No.*, the *Owner*, or the *Status*.
3. Click **Create** to create a new request.

#### ⓘ Note

Note that there can only be one default customizing request.

4. (Optional) Add business configurations to this request by, for instance, using the *Upload Business Configurations* app or the *Manage Number Range Intervals* app.

### Check All

1. Open the *Export Customizing Transports* app. A list of customizing requests is displayed, with the latest request shown first at the top.

2. You may put a filter on *Status* for modifiable and checking status. Additionally you can sort the transports in ascending order to see the current status of those transports sequentially.
3. Select *Check All > Start* to trigger consistency check for all open transports.
4. To check remaining open transports for which the consistency checks are yet to be performed, select *Check All > Show Progress*.

#### Note

Consistency check gets triggered in the background and the UI shows the latest transport request status as soon as *Check All* is triggered. The UI gets refreshed everytime after certain interval showing the latest status. The transport with status *Checking* shows the request currently being processed.

As soon as the consistency check is complete, a momentarily message gets displayed at the bottom.

If the *Check all* is already running, you can't trigger another and it will give you an error message

## Release Request

1. Release any tasks belonging to your transport request. A request can't be released until all its tasks have been released.
2. Check the consistency of your request by triggering a release simulation: Select the request and click *Release > Check*.
3. You can also release your request directly without triggering a simulation: Select the request and click *Release > Execute*.
4. To refresh the status of a released or simulated request: Select the request with the status *Checking* or *Release Started* and click *Refresh*.

Request No	Description	Type	Owner	Status	Transport Category	
ETOK901795	Business Configuration Project Implementation - Phase 1	Customizing Request	Example Configuration Expert - Business Process	 Checking	Default	>

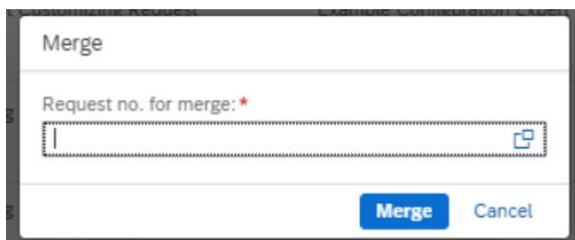
Request No	Description	Type	Owner	Status	Transport Category	
ETOK901795	Business Configuration Project Implementation - Phase 1	Customizing Request	Example Configuration Expert - Business Process	 Release Started	Default	>

5. View the results of the simulation or direct release in the *Logs* tab.

## Merge Request

If there are dependencies between different requests or several business users have recorded their configuration changes in different requests, you can choose to merge these different requests into one.

1. Select the request you want to merge.
2. Select *Merge*.
3. In the selection dialog, enter the target request number to which the selected request should be merged, and click *Merge*.



#### ⓘ Note

The configuration changes recorded in the request you selected to merge are copied to the target request. After merging, the previously selected request will be deleted.

## Create Copy

It can happen that for technical reasons, it's necessary for you to create a copy of an exported request. This can be the case if, for example, your exported request containing the business configurations didn't synchronize to the Git Repository. In order to synchronize these business configurations with the Git Repository again, a copy is needed.

1. Select the request you want to copy.
2. Click on [Create Copy](#).

## Repository Id

You can export the business configuration recorded in a request to a specific software component by assigning the [Repository Id](#) when creating a request.

1. Select [Create](#) to create a new request.
2. Using the value help, check the allowed repository ids which can be assigned to the request.
3. Select the suitable repository id and save the changes.

## 6.3.1.6.6 Maintain Customizing Translations

The [Maintain Customizing Translations](#) app provides an easy way to translate customizing extensions into other languages.

## Context

You've built a customizing extension and would now like to translate it into other languages. The [Maintain Customizing Translations](#) app allows you to select your text sources and generate an XLIFF (1.2) file containing

the translatable texts as well as useful notes on them. XLIFF (XML Localization Interchange File Format) is a standard used in localization. The XLIFF file can then be forwarded to a dedicated translator or translation agency. Once the XLIFF file has been translated, you can use the *Maintain Customizing Translations* app to upload it and publish the translated texts into your system.

## Key Features

You can use this app to:

- generate an XLIFF file of the texts that need to be translated.
- publish the translated texts to the system.

### ⓘ Note

If you want to have your texts translated by SAP, you can use the SAP Translation Hub. For more information, see

[SAP Translation Hub](#).

## Supported Device Types

- Desktop

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-DOC-TTL-MTR.

### 6.3.1.6.6.1 Create a Translation Project

## Context

You want to translate your customizing extension. Before you can select your text sources and target languages, you need to create a translation project.

### ⓘ Note

Please assign the business catalog SAP\_CORE\_BC\_BCT\_TRN\_MNG\_PC to ensure that changes can be recorded on transport requests.

## Procedure

1. Open the [Maintain Customizing Translations](#) app.
2. Click **+** (Create) to create a new translation project. Choose a project type from the dropdown.
3. Fill in an identifier for your project, a name, and a description.
4. Click **Create** at the bottom right of your screen.
5. You now need to add your translation project to a transport. You can create a customizing transport request by using the [Export Customizing Transports](#) app. For more information, see [Export Customizing Transports \[page 2582\]](#).

## Results

Your project has now been created and added to the overview list.

## Related Information

[Select Your Text Sources \[page 2588\]](#)

### 6.3.1.6.6.2 Select Your Text Sources

## Context

Select the text sources that you want to have translated.

## Procedure

1. In the [Maintain Customizing Translations](#) app, open the project you want to add text sources to.
2. Scroll down to **Text Sources** and click **Add**.
3. Here you can select which text source types to add. The following text source types are supported:
  - UI runtime adaptations

#### ⓘ Note

Mind that only the translation of those UI runtime adaptations which have been recorded on a Customizing transport request are supported. UI runtime adaptations which have not been recorded at all or have been recorded on a workbench request will not be considered

For more information on how to work with UI runtime adaptations, see [Adapting SAP Fiori UIs at Runtime - Key User Adaptation](#)

- SAP Fiori launchpad Pages
- SAP Fiori launchpad Spaces

**ⓘ Note**

For more information on how to manage launchpad pages and spaces, see [Manage Launchpad Pages](#) and [Manage Launchpad Spaces](#).

- Text tables

**ⓘ Note**

A database table needs to fulfill the following requirements to be considered a text table by the app:

- The delivery class of the text table has to be C
- There must be exactly one language key field
- Every character-like, non key field with a length greater than one will be treated as a text attribute
- The translation of text tables that contain key fields of type p (packed number), i (integer) or x (byte field) is not supported. That includes any kind of UUID and raw types, such as SYSUUID\_X16.

4. Select [Add](#) to open the [Add Text Source](#) dialog which displays all text sources that can be added to the translation project. By filtering the type and name, you can choose the text sources you want to add. Select [Add](#) to add the text sources you selected to your translation project.

**ⓘ Note**

The total amount of texts in a translation project must not exceed 1000. Be aware that a text source might contain more than one text.

## Related Information

[Select Your Source and Target Language \[page 2589\]](#)

### 6.3.1.6.6.3 Select Your Source and Target Language

## Context

Select your source language as well as the languages you want to translate your texts into.

## Procedure

1. In the *Maintain Customizing Translations* app, open the project for which you want to create a translation from a source language to a target language. You can choose to select either a single language or more than one language.
2. Scroll down to *Translations* and click *Create*.
3. Select a *Source Language* and *Target Language* from the dropdown menus and confirm with *Create*.

### ⓘ Note

You can select from all of the languages that are installed in the system. For a given text source, it's only possible to maintain translations for languages other than the original language.

Note that deleting a previously created translation entry for a given source and target language pair will only delete the corresponding entry in the translation project. Already published texts for the target language of the translation will not be deleted from the system.

4. A new entry has now been added to the list.
5. Repeat these steps for all source and target languages.

## 6.3.1.6.6.4 Generate and Download the XLIFF File

## Context

You can generate an XLIFF file for your source and target language(s) that you can download and send to your dedicated translator or translation agency.

## Procedure

1. In the *Maintain Customizing Translations* app, open the project you want to have translated and scroll down to *Translations*.
2. Select the entry for the source and target language for which you want to generate the XLIFF file.
3. Navigate to *Download*. Here you can select whether the XLIFF file should contain all texts (*All Texts*) or only those texts that were changed since you last published a translated XLIFF file in these languages (*Changed Texts*). If this is the first time you are generating an XLIFF file for the project for the given source and target language, choose *All Texts*.

The XLIFF file will now be downloaded to your computer. You can send it to your dedicated translator or translation agency to be translated.

## Related Information

[Upload and Publish the Translated XLIFF File to the System \[page 2591\]](#)

### 6.3.1.6.6.5 Upload and Publish the Translated XLIFF File to the System

#### Context

Your texts have been translated and a new XLIFF file has been sent back to you. You can now upload and publish the translated texts into the system. Here's how:

#### Procedure

1. In the [Maintain Customizing Translations](#) app, open your project and scroll down to [Translations](#).
  2. Select the entry for the source and target language for which you want to upload and publish the translated XLIFF file.
  3. Click [Upload](#).
  4. Search for the translated file and confirm with [Upload](#).
  5. Now the translations have been uploaded to the system. They still need to be published, however. Select the entry and click [Publish](#).
  6. Select a transport to which you want to add the corresponding language (LANG) transport entries and click [Select](#).
- Your translations have now been published.

### 6.3.1.7 Communication Management

The communication management apps allow you to integrate your system or solution with other systems to enable data exchange.

#### Prerequisites

- Predefined communication scenarios are available for different use cases, for example the integration for employee data. Decide which scenario you are going to use to create a communication arrangement.

## Process Overview



The communication management apps allow you to establish secure communication between your solution and other systems. The best practice to organize efficient data exchange is to proceed as follows:

1. Create communication users for inbound communication according to your needs using the [Maintain Communication Users](#) app.
2. Create a communication system using the [Communication Systems](#) app that represents the system you want to communicate with. If the selected scenario contains inbound services, you select the user for inbound communication that you have created earlier.
3. Create a communication arrangement using the [Communication Arrangements](#) app. You select the communication system that you have created earlier. The information about the user for inbound communication will be populated automatically. If several users for inbound communication exist for the system, you can select the appropriate one. Otherwise the first user in the list will be selected automatically. If the selected scenario contains outbound services, you have to enter the outbound user information manually.
4. Select [Save](#) to activate the communication arrangement.

### Note

A communication scenario is a design-time description of how two communication partners communicate with each other. It consists of inbound and/or outbound services as well as supported authentication methods. It provides technical information, such as the used inbound and outbound services and their service type, for example OData or SOAP, and the number of allowed communication arrangement instances. If the scenario exposes inbound services, it specifies the authorizations that are required to execute the services. The following types of communication scenarios are available:

- [Managed by SAP](#), where SAP provides a ready-to-use communication scenario and you create and maintain a communication arrangement.
- [Managed by Customer](#), where you develop a communication scenario and create and maintain a communication arrangement.

## Related Information

- [Maintain Communication Users \[page 2593\]](#)  
[Communication Arrangements \[page 2595\]](#)  
[Communication Systems \[page 2598\]](#)  
[IAM Information System \[page 2701\]](#)

## **6.3.1.7.1      Maintain Communication Users**

You can use this app to create and edit communication users. Communication users are used by solutions to authenticate themselves to be able to post data.

With this app you can manage communication users for different integration scenarios with other solutions. A communication user enables the integration with other solutions. To be able to post data, the solutions have to authenticate themselves with the user and password you create here. The communication users are assigned to the communication system you want to use.

### **Key Features**

You can use this app to:

- Create a user
- Edit a user
- Lock or unlock a user
- Delete a user
- Display communication systems that use the selected communication user
- Display communication arrangements for the systems that use the selected communication user
- Download a list of all communication users to a spreadsheet

### **Supported Device Types**

- Desktop
- Tablet

### **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### **Related Information**

[Communication Systems \[page 2598\]](#)

[How to Create Communication Users \[page 2594\]](#)

## 6.3.1.7.1.1 How to Create Communication Users

### Context

To create a new communication user, perform the following steps:

### Procedure

1. On the initial screen select [New](#).
2. Enter a valid user name and description.
3. Either provide a password (basic authentication) or upload certificates for certificate-based authentication.

#### ⓘ Note

We recommend that you use certificate-based authentication.

1. Configure password-based authentication

#### ⓘ Note

We recommend that you use the [Propose Password](#) button.

#### ⓘ Note

Passwords for communication users need to comply with the following requirements: the password length must be at least 20 characters, the password must contain at least one special character. The user will be locked after three unsuccessful login attempts.

2. Configure certificate-based authentication by uploading a X.509 client certificate. The customer system must use a client certificate signed by an appropriate certification authority (CA). A list of all root CAs approved by SAP Global Security is available in SAP Note [2801396](#) (SAP Global Trust List).
4. Select [Create](#) to save the user.

You can now assign the created user to a communication system. Use the [Communication Systems](#) app for this purpose.

### Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

## 6.3.1.7.2 Display Communication Scenarios

You can use this app to get an overview of available communication scenarios.

With this app you can display communication scenarios used for integrations, and the scenario status.

### Key Features

You can use this app to:

- Display all available communication scenarios
- Display scenario details and properties
- Display supported inbound authentication methods and inbound services used in a communication scenario
- Display supported outbound authentication methods, certificates, and outbound services used in a communication scenario
- Download certificates for specific purposes
- Display communication arrangements in which a communication scenario is used
- Create a new communication arrangement based on a communication scenario
- Display scope items: check which scope items the communication scenarios depend on. Communication scenarios that do not depend on any scope items are always visible in the system. For these communication scenarios, *Scope Items (0)* is displayed on the relevant tab and the following message appears in the table: *The communication scenario is not scope-dependent*.
- Display component: check under which component you can create an incident if required.

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

## 6.3.1.7.3 Communication Arrangements

With this app you can create and edit communication arrangements that your company has set up with a communication partner. The system provides communication scenarios for inbound and outbound communication that you can use to create communication arrangements. Inbound communication defines

how business documents are received from a communication partner, whereas outbound communication defines how business documents are sent to a communication partner. The communication scenario determines the authorizations, inbound and outbound services and the supported authentications methods, that are required for the communication.

## Key Features

You can use this app to:

- Display all existing communication arrangements
- Display detailed information for the selected communication arrangement
- Create new communication arrangements
- Maintain general information
- Select the required communication system. The user information and authentication method are filled in automatically after you have selected a communication system.
- Delete communication arrangements that you have created earlier

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

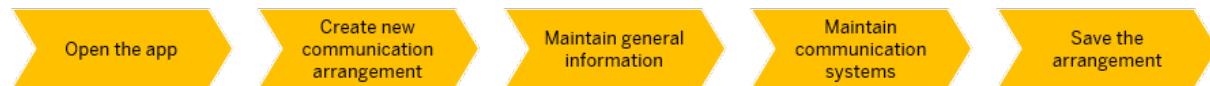
### 6.3.1.7.3.1 How to Create a Communication Arrangement

#### Prerequisites

- You have already created a communication system with inbound and outbound users.
- You have already created a communication user with a supported authentication type that is defined in the selected communication scenario.

## Context

### Process Steps



To create a communication arrangement, proceed as follows:

### Procedure

1. Open the *Communication Arrangements* app from the SAP Fiori Launchpad. Already existing communication arrangements are listed on the initial screen.
2. Select *New*. In the *New Communication Arrangement* window, select a communication scenario and define the arrangement name. Select *Create*.
3. The new *Communication Arrangements* screen is now open. In the *Communication System* field under *Common Data*, select a communication system that you want your system to connect to. Alternatively, you can click *New* to create a new communication system.
4. Depending on the underlying communication scenario, maintain the *Additional Properties*.
5. If the communication scenario provides inbound services, you have to select the desired communication user and authentication method from the assigned communication system. The defined communication user then has the authorization to call these services. In the *Inbound Services* section, the URLs to the service endpoints are displayed. Depending on the underlying scenario, maintain the required inbound parameters.
- Note:** Always use the API hostname of your tenant including the "-api" suffix. Don't use the UI hostname without the "-api".
6. If the communication scenario provides outbound services, you have to select the desired outbound user and authentication method from the assigned communication system. In the *Outbound Services* section, the URL path and port need to be defined. Depending on the underlying scenario, maintain the required outbound parameters and job execution details.

#### ⓘ Note

If the required outbound service is of the type *Outbound SQL Access*, you need to make the following additional settings:

- *Remote Database Schema Name*: Here you enter the schema in the remote system containing the objects to be accessed.
- If required click *Use Default*. This reads the *Remote Database Schema Name* from the Outbound SQL access service.
- *Remote Database Name*: The database in the remote system containing the objects to be accessed. Usually you can leave the field empty unless the remote system hosts multiple databases.

7. Save the arrangement.

## Related Information

[Maintain Communication Users \[page 2593\]](#)  
[Communication Systems \[page 2598\]](#)

### 6.3.1.7.4 Communication Systems

You can use this app to create communication systems. Communication systems are created to enable the communication among different systems.

With this app you can create new communication systems that you can later use to establish communication arrangements. To enable communication between different systems you have to register these systems in the *Communication Systems* app. The communication system represents the communication partner within a communication. For inbound communication, this is the system that calls services provided by your system. For outbound communication, this is the system that provides services called by your system.

## Key Features

You can use this app to:

- Create communication systems
- Display detailed information about the already existing communication systems
- Delete communication systems

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

## Related Information

[Maintain Communication Users \[page 2593\]](#)

## 6.3.1.7.4.1 How to Create Communication Systems

### Context

To create a communication system perform the following steps:

### Procedure

1. On the initial screen of the *Communication Systems* app, select *New*.
2. In the *New Communication System* dialog define an ID and a name of the new system.
3. Fill in the required fields under:
  - *General Data*  
Enter system ID and system name.  
*Contact Information*  
Enter name, email or phone number of a contact person.
  - *Technical Data*  
*General*  
If you want your communication system to be used for a communication arrangement based on a communication scenario that only contains inbound services, we recommend that you select the *Inbound Only* checkbox. If this checkbox is selected, the input fields that are only required for outbound communication disappear.  
If your communication system is intended for outbound communication, host, port, user and authentication method are used as specified in the destination service.
  - *Destination Service* (Only for Outbound Communication)  
Because SAP BTP for the ABAP environment uses destination services of SAP BTP for outbound communication, you need to enter the name of the required destination service. For more information, see [Set Up the Destination Service](#). You also need to select the instance the destination service is based on. For more information, see [Creating the Service Instance for the ABAP Environment](#).

#### ⓘ Note

A predefined instance is selected by default. If you want to add your own instance, deselect this instance.

- *Cloud Connector* switch  
Indicates that SAP Cloud Platform Cloud Connector is used for communication with this system.  
*SSC Location ID*: Enter the SAP Cloud Platform Cloud Connector ID  
If your scenario is uses RFC modules, and SAP Cloud Platform Cloud Connector is not activated, the system uses WebSocket RFC. Therefore the remote system needs to support WebSocket RFC and you need to specify the client in the *RFC Settings* section.
- *RFC Settings*  
Define additional settings needed for RFC communication.

- [OAuth 2.0 Settings](#)

Define additional settings if oAuth 2.0 is used for outbound communication.

 Note

The propagation of technical users from the cloud application towards on-premise systems can be enabled in the [Communication Systems](#) app. To propagate the technical user, you have to select the [Cloud Conn. Technical User Propagation](#) checkbox in the [OAuth 2.0 Settings](#) area (the checkbox is only active if the [Cloud Connector](#) switch is on). This is similar to principal propagation, but in this case, a technical user is propagated instead of a business user.

- [Remote SQL Access](#) switch

Define additional settings needed to access remote databases via SQL:

Select the required [Adapter Name](#). This specifies the corresponding HANA Smart Data Access adapter and the type of access method to be used by the SAP HANA database to access the data. Use the field input help for the list of adapter names you can specify.

Enter the required [Adapter Configuration](#). This specifies the connection parameters for a given adapter.

- [OAuth 2.0 Identity Provider](#) switch

Before you can authenticate and get an access token to access resources using an OAuth 2.0 client, you have to configure a trusted relationship to the required [SAML Identity Provider](#) (token issuer).

Upload the signing certificate of the trusted provider and define the provider name.

Define the [User ID Mapping Mode](#) if you don't use e-mail addresses, but either logon name or global user ID.

The [User ID Mapping Mode](#) is required when the [SAML Identity Provider](#) is creating SAML assertions with the [NameIdentifierFormat unspecified](#).

This setting is ignored when then SAML assertion is using the [NameIdentifierFormat eMailAddress](#).

Please note that the required authorization for calling the service is granted to the business user and not to the technical user. When you set up a communication arrangement using oAuth, you can click [OAuth 2.0 Details](#), select the scope and click [Granted by Business Catalogs](#) to find the necessary details. This way you can retrieve the business catalogs that are required for calling the service.

- [SAML Bearer Assertion Provider](#)

Before you can authenticate using SAML2 Bearer assertions transmitted via the authorization header, you have to configure a trusted relationship to the required [SAML Identity Provider](#) (token issuer).

Upload the signing certificate of the trusted provider and define the provider name.

Define the required [User ID Mapping Mode](#) if it is unspecified and you don't use e-mail addresses, but either logon name or global user ID.

The [User ID Mapping Mode](#) is required when the [SAML Identity Provider](#) is creating SAML assertions with the [NameIdentifierFormat unspecified](#).

This setting is ignored when then SAML assertion is using the [NameIdentifierFormat eMailAddress](#).

- [OpenID Connect](#) switch

Before you can authenticate using OpenID Connect, you have to configure a trusted [OpenID Connect Provider](#).

Define the required [User ID Mapping Mode](#), the [OIDC Token Issuer](#) and [Client ID](#). Click [Import Metadata](#). This is necessary to ensure that the configuration is validated and complete.

If the automatic configuration fails, you can configure the OpenID Connect settings manually. For more information, see [Manual Configuration - OpenID Connect \(OIDC\) Provider](#) in the [Related Information](#) section.

**ⓘ Note**

You can select the [User Mapping Claim](#) to define the authorization attributes that are used by the identity provider. The default selection is *sub*.

4. Add technical users for inbound communication. You can either select a user from the list or create a new user. If you decide to create a new user, you will be redirected to the [Maintain Communication User](#) app.

**ⓘ Note**

Inbound users are communication users provided by your system and are used by the communication system to call the inbound services.

5. Save the changes.

You can now establish a communication arrangement with the created system. Use the [Communication Arrangements](#) app for this purpose.

## Related Information

[Communication Systems \[page 2598\]](#)

[How to Create Communication Users \[page 2594\]](#)

[Manual Configuration - OpenID Connect \(OIDC\) Provider \[page 2603\]](#)

### 6.3.1.7.4.2 Authentication Methods for Outbound Users

You can define how your solution authenticates itself when it sends business documents to the external system. For security reasons, we recommend the use of certificate-based authentication.

You can choose between the following authentication methods:

- User name and password
- SSL Client Certificate
- OAuth 1.0
- OAuth 2.0

You can choose between the following authentication methods:

- Basic  
If you use this method, you need to enter a client secret.
- mTLS  
If you use this method, you need to select a client certificate.
- JWT  
You can use JWT (JSON Web Tokens) to authenticate an OAuth client at the token endpoint of an OAuth authorization server (instead of using a client ID and a client secret).  
If you use this method, you need to download the corresponding certificate to your local device.
- None

If you want to use a SSL Client Certificate, you can either use the default certificate of your solution or a client certificate you have defined in the [Maintain Client Certificates](#) app.

## Related Information

[Maintain Client Certificates](#)

### 6.3.1.7.4.3 Enabling Public OAuth Clients (PKCE) for Inbound Communication Scenarios

#### Context

Communication scenarios that support OAuth 2.0 as authentication method for inbound communication can now be consumed by public OAuth clients using PKCE (Proof Key for Code Exchange). This allows desktop applications (acting as OAuth Clients) to consume OAuth-protected OData services with the identity of a business user. The business user will be prompted for providing consent to this action once (a web browser is required for this).

To enable consumption by public OAuth clients using PKCE, proceed as follows:

#### Procedure

1. In the [Communication Systems](#) app, enter the [Client Redirect URI](#) under Inbound OAuth 2.0 Client Settings. Thereby you define the URL to which the client will redirect the browser after authorization has been granted by the user.
2. Under [Users for Inbound Communication](#), click Add.
3. In the [New User for Inbound Communication](#) pop-up that appears, select [OAuth 2.0](#) in the [Authentication Method](#) dropdown list, enter the [OAuth 2.0 Client ID](#) and click [OK](#).

## Related Information

[How to Create a Communication Arrangement \[page 2596\]](#)

## 6.3.1.7.4.4 Manual Configuration - OpenID Connect (OIDC) Provider

When you have made the required settings under [OpenID Connect \(OIDC\) Provider – Basic Configuration](#) in the [Edit](#) mode of the communication system, the system attempts to automatically configure OpenID Connect for the provider you have defined. If the provider does not convey the required information, you receive an error message. In this case, you can use manual configuration as a fallback. Activate the [Manual Configuration](#) switch if this is required.

As a first step, you can download web keys from the provider. Should this fail too, you can directly enter the web keys as defined in your OpenID Connect specification.

### Related Information

## 6.3.1.7.5 Maintain Extensions on SAP BTP

You can use this app to create [SAP Business Technology Platform \(SAP BTP\)](#) extensions to build extension applications.

With this app you can create SAP BTP extensions for your SAP S/4HANA Cloud system. These automatically connect the two systems and enable you to build extension applications for SAP S/4HANA Cloud on SAP BTP.

### Key Features

You can use this app to:

- Create new SAP BTP extensions
- Temporarily disable SAP BTP extensions

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

## Related Links

- [Trigger the Registration in the SAP S/4HANA Cloud Tenant](#)

### 6.3.1.7.6 Display Inbound Services

With this app you can display all available inbound services defined by you or delivered by SAP. You can also display further information about the services, such as if and where a deprecated inbound service is used and a list of its successors. You can also check whether the service is deprecated system-wide or only in a scenario.

## Key Features

You can use this app to:

- Display documentation for the service
- Display authorization objects (this is only relevant in the custom communication scenarios environment)
- Display successors
- Display all communication scenarios that use this service

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.7.7 Display Outbound Services

With this app you can display all available outbound services defined by you or delivered by SAP. You can also display further information about the services, such as if and where a deprecated outbound service is used and a list of its successors. You can also check whether the service is deprecated system-wide or only in a scenario.

## Key Features

You can use this app to:

- Display documentation for the service
- Display successors
- Display all communication scenarios that use this service

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.7.8 Business Catalogs for Communication Management

You assign business catalogs to business roles that are assigned to business users. Business catalogs contain authorizations that define what a business user with a certain business role is allowed to do.

Business Catalogs for Communication Management Apps

Business Role Catalog	Authorizations	Restrictions
Communication Management	All authorizations for the COM apps	None
SAP_CORE_BC_COM		

## 6.3.1.7.9 Glossary for Communication Management

Overview of terminology used for communication management

Terminology Overview

Term	Description
Communication arrangement	A communication arrangement describes a communication scenario with a remote system during configuration time. It provides the necessary metadata for service configuration.
Communication system	A communication system represents the communication partner with all technical information that is required for communication, that is hostname, identity, user information, certificates, etc.
Communication user	A communication user is a special type of technical users that is assigned to a communication system. You create a communication user for particular communication scenarios.
Communication scenario	A communication scenario defines the communication between external systems and your own SAP cloud system. It bundles inbound and outbound services and additional properties that are required for configuration settings.

## 6.3.1.7.10 Display Connectivity Trace

With this app you can analyze inbound connectivity issues, such as failed SSL handshakes, malformed HTTP requests or failed log-in.

### Key Features

You can use this app to:

- Display connectivity trace details
- Cross-navigate to the *Maintain Business Users* app and the *Maintain Communication Users* app.
- Maintain trace criteria

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.7.10.1 How to Create a Trace

#### Context

In order to start a trace, you need to define the trace criteria.

#### Procedure

1. Select the *Display Connectivity Trace* tile on the SAP Fiori Launchpad to open the app. On the initial screen, click *Define Trace Criteria* in the top-right corner.
2. Click *Create*.
3. Enter a description if required. We recommend using a description that is easily identifiable in the filter.
4. Change the validity if required. The default setting is *10 Minutes*. Unlike a log, a trace is only available for a limited time frame.
5. Change the *Request Method* if required. The default setting is *All*.
6. Click *Save*.

#### Results

The trace is now activated for the defined criteria. You can filter for it in the *Trace Criterion* dropdown list. Please note that there might be a time delay until the activation takes effect because all the affected components need to be notified.

## 6.3.1.7.11 Monitor bgRFC Queues

With this app you can monitor the bgRFC (background RFC) queues together with the associated units and inbound destinations. As well as displaying queues and units that belong to a destination, the app provides you with functions that let you intervene in their processing, such as *Stop*, *Start*, or *Delete*.

### Key Features

You can use this app to:

- Stop and start destinations, queues, units
- Delete queues and units if they are no longer needed
- Check for errors
- Display the first function module for each unit
- Navigate to Message Monitoring to view more details if required

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### Related Information

## 6.3.1.7.11.1 Monitor bgRFC Queues: Further Information

Please note the following when using the *Monitor bgRFC Queues* app.

- Deleting a queue or unit can be risky. If you delete a queue, the included units are automatically removed. This can cause issues in the processing because dependencies between units and queues might exist.
- Queues and units can be stopped from the monitor. This function is only intended for analysis and not for regular operations, because it has consequences for runtime. If a queue is to be stopped, this also refers to the top unit that uses this queue. As soon as the unit is restarted, the queue is released again.

- Transactional (*T-Type*) units can be displayed to check whether any errors occurred. They are listed below the *BGPF* (Background Processing Framework) *Inbound Destination*. To call them up, you need to click the *Standard* list entry of the relevant queue.
- In the *Queue* table, the *Start All* and *Delete All* pushbuttons enable batch operations. When clicked, they apply their respective actions to all current entries in the table based on the filters that have been set. Therefore, if no filters are in place, all queues in the table will be affected. Conversely, if filters have been set, only those queues matching the filter criteria will be impacted. If you click one of these pushbuttons, a confirmation dialog box appears indicating the number of queues that will be impacted by the action.

### **6.3.1.7.12 Maintain SAML 2.0 Configuration**

You can use this app to upload the primary and secondary signing certificates you have maintained within the Cloud Identity Services. Please note that Base64 ASCII encoding is required for this.

#### **Key Features**

This app provides the following key features:

- Upload primary signing certificates
- Upload secondary signing certificates
- Delete secondary signing certificates
- Display detailed information for the certificates, such as validity, subject, or issuer

#### **Supported Device Types**

- Desktop
- Tablet

#### **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### **6.3.1.8 Custom Code Migration**

#### **Purpose**

The Custom Code Migration app enables you to create three different kinds of projects:

1. SAP S/4HANA Migration Project:  
Analyze custom code that shall be migrated from an existing product like SAP Business Suite to a new product such as SAP S/4HANA 2023. To evaluate the custom objects to be adapted, it performs the SAP S/4HANA custom code checks.
2. SAP BTP Analysis Project:  
Analyze custom code for readiness to run in SAP BTP, ABAP environment.
3. Custom Code Analysis Project:  
Analyze custom code with arbitrary ATC check variants. See [Performing an Analysis \[page 2615\]](#).

In addition, this app supports you with identifying unused custom code based on your collected usage data. This enables you to remove unused custom code during a system conversion to SAP S/4HANA.

## Key Features

Analysis of custom code:

- Graphical representation of custom code analysis results
- Results can be filtered by various categories, such as quick fix availability, scope and usage data.
- Options to handle findings after analysis (see [Process Findings After Analysis \[page 2620\]](#))

Scoping:

- Based on usage data, you can define the ABAP custom code that needs to be migrated to SAP S/4HANA.
- This app creates a deletion transport in order to delete unused ABAP source code during the system conversion to SAP S/4HANA.

## Access Information

For more information on how to enable this app, see [Enable Usage of the Custom Code Migration App \[page 2611\]](#).

## Component for Customer Incidents

BC-DWB-CEX

## Supported Device Types

- Desktop

## 6.3.1.8.1 Enable Usage of the Custom Code Migration App

### Prerequisites

- Your company has a global account for *SAP Business Technology Platform*.
- You need permission to access the *Cloud Connector*.

### Procedure

#### Maintain Business Role

To enable business users to access the *Custom Code Migration* app, create the *Project Manager – IT* business role and assign one or more business user(s) to it. Please note that, in order to use the feature of adding findings to the baseline, you'll also need the *Quality Manager – Software Development* business role. To create these roles, proceed as follow (taking *Project Manager – IT* as an example):

1. From the SAP Fiori launchpad of your ABAP environment, open the *Maintain Business Role* tile.
2. Create the *Project Manager – IT* business role using the *SAP\_BR\_IT\_PROJECT\_MANAGER* template.  
For more information see, [How to Create a Business Role from a Template](#).
3. Maintain the restrictions of the business role.

#### ⓘ Note

To enable write access for this business role, ensure that it is set to *Unrestricted*.

4. Open the new business role and add the *Assigned Business Users*.
5. *Save* the changes.

#### Maintain Communication Arrangement

To enable communication from your ABAP environment to your on-premise systems using Remote Function Calls (RFC), you need to create the communication arrangement *SAP Custom Code Migration Integration* (*SAP\_COM\_0464*) in your ABAP environment.

To set up the connection from the ABAP system in the ABAP environment to your on-premise system, proceed as follows:

1. Log on to the *SAP Fiori launchpad* of your ABAP environment.
2. In the *Communication Management* section, select the *Communication Arrangement* tile.
3. Create a new communication arrangement using the *SAP\_COM\_0464* scenario.
4. If not yet available or defined, create a communication system for the communication arrangement to define an endpoint for your checked system.
  1. For the *Communication System*, choose *New*.
  2. In the *New Communication System* dialog, enter the *System ID* and *System Name* of the checked system.
  3. Choose *Create*.

4. In the *Technical Data* tab under *General*, enter the virtual host as specified in your *Cloud Connector* as *Host Name*. The field *Port* has already been filled in automatically with the default **443**.
5. Switch on the slider for *Cloud Connector*.
6. Filling in the field *SCC Location ID* is optional.
7. Under *RFC Settings*, fill in the fields *Client*, *Instance Number* and *Target Host* as specified as virtual host in your *Cloud Connector*.
8. Under *User for Outbound Communication*, choose **+** to assign the RFC user you created in the checked system to the communication system. See [Configuring Remote ATC Using a Central Check System](#) to learn how to create such an RFC user.
9. Choose *Create* and then *Save* to save the communication system and to be navigated back to the Communication Arrangement creation page.
10. Confirm with *Save*.
5. Under *Additional Properties*, fill out the fields *Object Provider* and *System Group*. An object provider defines the RFC connection to be used for analysis in a remote SAP system and must be assigned to a system group. This is the name you then select as your *Object Provider to Remote System* for your custom code migration project in the *Custom Code Migration* app. A system group subsumes multiple SAP systems (typically, the productive system, the test system(s), and the development system(s)), which all represent a part of a system landscape of one and the same SAP release.
6. Under *Outbound Services* *Retrieve Custom Code*, ensure that the *Service Status* is set to *Active*.
7. Under *Outbound Services* *Display Custom Code*, ensure that the *Service Status* for displaying custom code via UI link is set to *Active*.
8. Choose *Save* to save the communication arrangement. A message should now pop up at the bottom of the screen informing you that the activation was successful.

You can now select the communication arrangement in the *Object Provider to Remote System* field in the *Custom Code Migration* app to establish the connection to your on-premise system.

## On-Premise System: Install Remote Stubs

To analyze your custom code in your on-premise system using the *Custom Code Migration* app, see SAP Note [2599695](#).

## Related Information

[SAP Note 2436688](#)

[Custom Code Migration Guide for SAP S/4HANA](#)

[How to check your custom ABAP code for SAP BTP, ABAP environment](#)

## 6.3.1.8.1.1 Configuration

You can perform remote code analysis in ATC which allows to analyze custom code remotely with the latest checks.

When using SAP BTP ABAP environment as central check system, you will use the `Custom Code Migration` app to scope and analyze custom code in an on-premise SAP system for SAP S/4HANA Cloud Public Edition.

To use this SAP Fiori app, your SAP administrator must enable the app provide access for the relevant users. In addition, you also need to establish an RFC connection for the Cloud Connector. This enables you to access the on-premise system as checked system from the ABAP environment as central check system remotely.

To find out more about this configuration, see [Configuring Remote ATC Using a Central Check System](#).

## 6.3.1.8.1.2 Administrating User Assignments

The user assignment describes the relevant roles that a user requires to access SAP BTP ABAP environment using the `Custom Code Migration` app.

### Context

You want to enable business users to access the `Custom Code Migration` app.

For more information see, [How to Create a Business Role from a Template](#).

#### ⓘ Note

If you use `Prepare an Account for ABAP Development booster`, you can skip this step. The booster will perform this task.

### Procedure

1. In the SAP Fiori launchpad, navigate to the [Identity and Access Management](#) section and open the `Maintain Business Roles` app.
2. On the initial screen, select [Create from Template](#) from the bottom of the page.  
The [Create Business Role from Template](#) wizard is opened.
3. Use the value help to choose `SAP_BR_IT_PROJECT_MANAGER` as the underlying *Template*.
4. Enter `BR_IT_PROJECT_MANAGER` as the name of the *New Business Role ID*.
5. Enter a *New Business Role Description*.
6. The checkbox *Create and Assign Spaces Based on SAP-Delivered Spaces* provides you with the following two options:
  - a. **Mark** the checkbox to create a new space for the `Custom Code Migration` app in your subaccount. Define and enter a *New Space ID*.

- b. **Unmark** the checkbox to use an existing space for the CCM app. Consequently, you must allocate the Custom Code Migration app to the existing space and page in your subaccount. For more information, see [Working with Predefined Spaces and Pages](#).
7. Confirm the creation with *OK*.  
The new business role is created and opened in the *Maintain Business Role* page.
8. Select *Maintain Restrictions* from the title toolbar.  
The relevant subpage is opened.
9. In the *Write, Read, Value Help* tab, choose *Unrestricted* from the drop-down list to enable write access for this business role.
10. Navigate back to the main page of your new business role.
11. Open the *Assigned Business Users* tab and add the relevant users using the *Search* field or the *Add* button.
12. *Save* the changes from the bottom.

## Results

The business role is created and added. The assigned users now have access to the Custom Code Migration app.

### 6.3.1.8.2 Custom Code Analysis

This guide describes how to perform remote custom code analysis using the Custom Code Migration app.

The Custom Code Migration app in the SAP BTP ABAP environment can act as the ABAP Test Cockpit (ATC), a central check system analyzing the ABAP source code located in your on-premise system. It enables you to identify functional or security vulnerabilities in your ABAP source code. This includes security checks performed by the SAP Code Vulnerability Analyzer (CVA). Further cloud use cases for ATC in the Custom Code Migration app involve the migration to S/4HANA, performance checks, and ABAP Cloud readiness checks.

In the following chapters, you will learn how to create a custom code analysis project, analyze the findings, and, in addition, perform dependency and complexity analyses.

## Related Information

[Code Vulnerability Analyzer Checks](#)

## 6.3.1.8.2.1 Performing an Analysis

This chapter focuses on creating a custom code analysis project, analyzing the findings and how to perform a dependency and complexity analysis.

### 6.3.1.8.2.1.1 Creating a Custom Code Analysis Project

With a custom code analysis project, you can analyze custom code with arbitrary ATC check variants.

#### Procedure

1. Start the tile *Custom Code Migration* from the SAP Fiori launchpad.
2. Choose *Create*, then choose *Custom Code Analysis Project*.
3. Fill in the fields under *Project Data* for your new project.
  - a. *Project Description*: Provide further details about your custom code migration project.
  - b. *Check Variant*: Use the value help to select a check variant available in your system.
  - c. *Location of Custom Code*: Use the dropdown menu to specify the system in which the custom code to be analyzed is located.
  - d. *Object Provider to Remote System*: An object provider defines the RFC connection to be used for analysis in a remote SAP system and must be assigned to a system group.

#### ⓘ Note

If you collected usage data, choose the *Add* button to add usage data to your custom code migration project.

Under *Advanced Configuration*, you can limit your custom code analysis by specifying the packages you want to analyze. This can be helpful to save time and resources.

These steps are optional.

4. In the *Advanced Configuration* tab, under *Analysis Configuration*, you can either leave the check box checked and start the custom code analysis automatically at project creation or uncheck it and start the analysis manually at a later point.
5. To confirm your input, choose *Create*.

## 6.3.1.8.2.1.2 Analyzing the Findings

When you create a custom code analysis project, the custom code analysis is performed automatically. The `Custom Code Migration` app then provides you with an analytical representation of the findings.

### Procedure

1. Choose [Analysis](#) to get an overview of the results, as soon as the first analysis has been finished.
2. Choose [Analyze Findings](#) to get a detailed analysis.
3. In the [Custom Code Migration - Analysis - Findings](#) view, you can specify various filters, such as:
  - [Quick Fix Availability](#): Analyze which findings can be solved by a Quick Fix and which findings have to be solved manually.
  - [Scope Information](#): Filter the results to only show findings for objects that are in scope.
4. Choose  in the [Findings](#) section to change the settings for the chart.

For example, to show the Quick Fix availability per Check Message, you need to specify the following settings for the chart in the [Chart](#) tab under [Dimensions](#):

- Check Title (as Category)
- Check Message (as Category)
- Quick Fix Availability (as Series)

### Results

You now have a graphical and tabular representation of findings and can take action to resolve these findings in your development environment.

## 6.3.1.8.2.1.3 Dependency Analysis

In this topic, you will learn more about dependency analysis in the Custom Code Migration app.

### Purpose

You can utilize dependency analysis to identify dependencies from your custom code to the SAP standard and thus which part of your custom code can be transformed to the SAP BTP, ABAP Environment. To do this, you can use the [Scoping](#) app within the Custom Code Migration app. This app provides you with a set of measures that analyze how likely it is for you to be able to transform your code to the SAP BTP, ABAP environment.

## Note

Please be aware that as of the SAP BTP, ABAP environment 2411 release, you now need to initiate scope calculation manually after project creation. Scope calculation no longer starts automatically at project creation.

This is how you start a dependency analysis in the Scoping app:

1. In your project, select the [Scope](#) tab in the lower half of the project overview page.
2. Select [Change Scope](#).
3. Select the second [Packages](#) tab from the left. This tab is marked with a bubble chart icon.
4. Select the [Settings](#) () button.
5. Under [Measures](#), choose your metrics for the chart axes.

## Tip

This way you can set up your own individualized chart to perform dependency analysis of your custom code with SAP standard. You can determine, for example, how often your custom code calls SAP application APIs or uses DB accesses to SAP application tables such as the `MARA` table.

The following measures are available:

The measures and their meanings

Measure	Explanation
Number of Changes	Number of changes per object within the last year
Number of application API usages	All usages of application APIs such as classes, interfaces, and function modules
Number of application DB API usages	All database operations on database tables or views; i.e., only the number of different database tables and views are counted, not the overall number of accesses
Number of distinct app. DB APIs used	Database operations on distinct database tables or views
Number of distinct application APIs used*	Distinct usages of application APIs; i.e., only the number of different APIs are counted, not the overall number of accesses
Number of objects in package	Overall number of custom code objects within a package or request entry point
Number of objects using appl. DB APIs	Number of objects within a package or request entry point accessing application-specific database tables or views
Number of objects using application APIs	Number of objects within a package or request entry point using application APIs
Proportion of objects using app. APIs	Ratio between objects using application APIs and all objects within the package/request entry point

Measure	Explanation
Proportion of objects using app. DB APIs	Ratio between objects using application-specific database tables or views and all objects within the package/request entry point

\*With this measure, for example, you can check how many dependencies to different APIs your code has. In the best case, both this number and the number of locations where you are accessing those APIs are small.

After adjusting your settings, your chart could look as follows:



You can then also save your individual settings as your own view. This way, you won't have to adjust your settings again the next time you want to use the chart.

## Related Information

[Custom Code Migration \[page 2609\]](#)

[Complexity Analysis \[page 2619\]](#)

## 6.3.1.8.2.1.4 Complexity Analysis

In this topic, you will learn more about complexity analysis in the Custom Code Migration app.

### Purpose

Sometimes, your ABAP code can contain complex objects that very often also experience the most frequent changes. With the Custom Code Migration app it is possible to identify the most complex objects via complexity analysis and help you decide whether you want to redesign and therefore simplify them.

#### ⓘ Note

Please be aware that, in this topic, the focus is on reducing the complexity of packages only. Nevertheless, generally, what applies to packages also applies to request entry points.

In your Custom Code Migration project, choose the bubble chart to display all packages that contain custom code. With this chart, you can also display the Halstead difficulty of a package. The Halstead complexity measures are software metrics that describe the effort to develop and understand the complexity of a program.

The chart enables you to identify packages that contain complex coding and have often been changed in the last year, for example.

The default representation of the chart displays the following information:

The X axis indicates the complexity/difficulty of the custom code within a package. The higher the number signifies the complexity. The app calculates the complexity based on the Halstead complexity measures. For packages, the Halstead difficulty is aggregated using an H-index algorithm.

#### ⓘ Note

Only if you have multiple complex objects in your code does the H-index rise.

An H-index of 42 means that the package contains 42 procedures with a Halstead difficulty of 42 or higher.

An H-index of 80 means that the package contains 80 procedures with a Halstead difficulty of 80 or higher.

The Y axis displays how many changes – based on transport requests – you have made to the objects of a package.

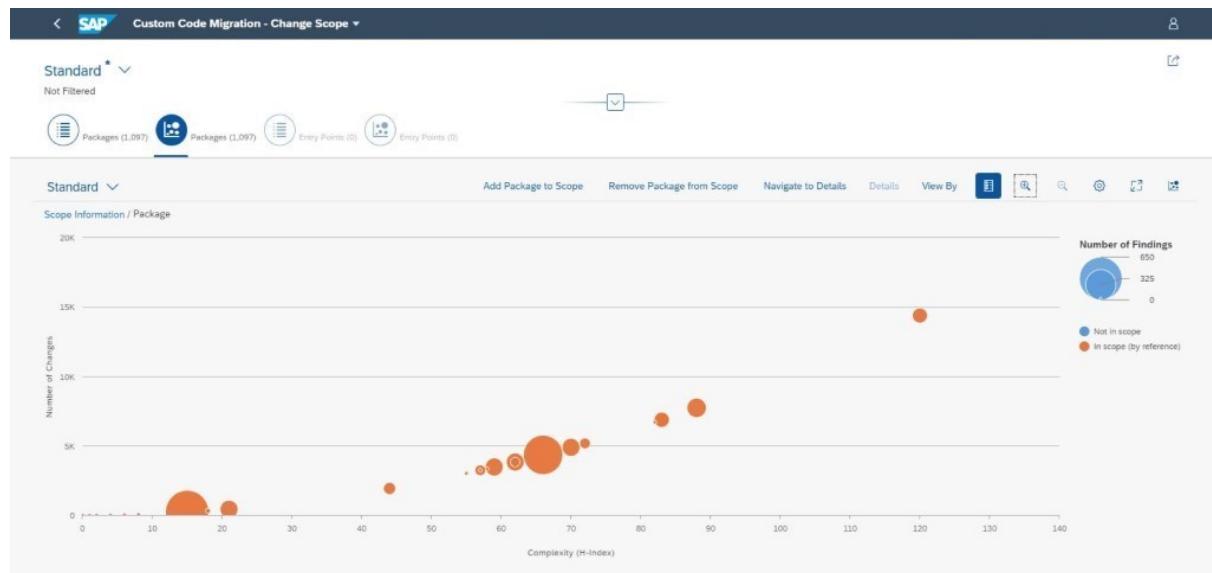
#### → Recommendation

SAP recommends you have a closer look at packages that appear in the top-right corner of this chart, as those are the packages that are very complex and have often been changed.

The size of the bubbles indicates the number of findings. Hence, the bigger the bubble, the higher the number of findings.

You can change the default representation of the chart by choosing the chart preferences (⚙).

Your chart could now look as follows:



Now that you are aware of the complexity of your code, it is time to take action and reduce said complexity. There is no standard solution for fixing complex code, however, SAP can provide you with suggestions as to how to start this process:

- Check whether you can replace your code with SAP standard functionalities.
- Check whether there are smaller units in your code that you can reuse.
- Check out the [Clean ABAP](#) style guide to learn more about clean code.

## Related Information

[SAP Note 2436688](#)

[Custom Code Migration Guide for SAP S/4HANA](#)

[How to check your custom ABAP code for SAP BTP, ABAP environment](#)

## 6.3.1.8.2.2 Process Findings After Analysis

Find out how you can process your findings after custom code analysis.

### Purpose

In case you've been wondering what the next steps would be after custom code analysis and what to do with your findings in the Custom Code Migration app, this topic is for you. Here, you will learn about your options for after custom code analysis.

## 1. ABAP Test Cockpit (ATC) Baseline

With the baseline functionality, it's now possible to add/remove custom code project check results to/from the baseline and choose whether you would like to:

1. Exempt your findings: ATC findings from the baseline will be marked as exempted in the check results list.
2. Suppress your findings: After a new ATC check run, all ATC findings from the baseline are excluded from the ATC run results list.
3. Reduce findings' priority: The priority will be reduced to low (priority 3) for all ATC findings of the relevant ATC check result.

**Result:** After adding findings to the baseline, they won't appear in the subsequent check results of the custom code project.

### ⓘ Note

Please be aware that:

- Once you have added findings to the baseline, the custom code migration project cannot be deleted. Remove your findings from the baseline first and then proceed with deleting your project.
- The analysis cannot be started again after the findings were added to the baseline.

## 2. SAP Readiness Check

You can download the check results to upload them later onto SAP Readiness Check 2.0. To do so, in the *Analyze Findings* section of your project, choose the arrow next to the spreadsheet icon and select *Export for SAP Readiness Check*. To learn more about SAP Readiness Check for SAP S/4HANA, see SAP Note [2913617](#).

### 6.3.1.8.3 Schedule Custom Code Analysis

Learn more about the *Schedule Custom Code Analysis* app.

#### Purpose

The *Schedule Custom Code Analysis* app enables you to schedule projects created in the *Custom Code Migration* app as application jobs. With this functionality, the analyses can be performed once or periodically. The *Schedule Custom Code Analysis* app is available to all users that have access to the *Custom Code Migration* app and have been assigned the *Project Manager – IT* business role.

#### Key Features

With this app you can schedule ABAP Test Cockpit (ATC) mass runs for your custom code migration project in a SAP BTP, ABAP environment system through a predefined custom code analysis application job template.

## Access Information

For more information how to provide access to users and how to implement this app, see [Enable Usage of the Custom Code Migration App \[page 2611\]](#).

## Component for Customer Incidents

BC-DWB-TOO-ATF

## Supported Device Types

- Desktop

### 6.3.1.9 Development

[SQL Trace Analysis \[page 2622\]](#)

[Display Publishing Processes \[page 2623\]](#)

#### 6.3.1.9.1 SQL Trace Analysis

With this app you can access and analyze SQL trace records after you have used the SQL trace function of the ABAP performance trace in the ABAP Development Tools (ADT).

## Key Features

This app provides the following key features:

- Display of all SQL traces that are available under your user name
- View of the SQL trace entries in chronological order, together with information about the ABAP source code location, duration of the execution, and the number of accessed records
- View and recalculation of the prepared and executed access plan
- Display of the origin of SQL statements in the source code

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-OP-MON.

### **6.3.1.9.2 Display Publishing Processes**

With this app you can monitor publishing processes for custom communication scenarios and business catalog extensions.

## **Key Features**

You can use this app to:

- Filter for a certain status, such as *Successful*
- Filter for a certain object type, such as *Custom Communication Scenario*
- View details of the publishing process
- Restart a publishing process if required

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## 6.3.1.10 Employee Master Data

[Business User Management \[page 2624\]](#)

Business User Management enables and supports the lifecycle maintenance of business users.

[Maintain Employees \[page 2624\]](#)

### 6.3.1.10.1 Business User Management

Business User Management enables and supports the lifecycle maintenance of business users.

### 6.3.1.10.2 Maintain Employees

With this app, you can create employees and modify employee information.

#### Key Features

You can use this app to:

- Create employees directly in the app or with CSV file import
- Edit employee information
- Retrieve the change log to view modified employee data
- Select employees for archiving
- Export employee data

In addition, the app supports the following technical features and options:

Navigate to [Maintain Business User](#)

#### Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component CA-GTF-BUM.

### 6.3.1.10.2.1 How to Select an Employee for Archiving

#### Context

You can't delete an employee in the *Maintain Employees* app. You can only select an employee for archiving. This is contrary to the *Maintain Business Users* app, where the user is physically deleted.

##### Note

It's not possible to select an employee for archiving in the *Maintain Employees* app if the employee has a user assigned to it. In this case, you have to delete the user in the *Maintain Business Users* app first.

The employee consists of a business partner in the role category BUP003 (Employee), a workplace address, and a user. An employee cannot be deleted immediately and in one step for business process reasons. If you need to permanently delete an employee, you have to delete the user first. Afterwards, you're able to delete the business partner, which also contains the workplace address. Therefore, to physically delete an employee you need to refer to your Information Lifecycle Management (ILM) apps.

In the *Maintain Employees* app, the *Select for Archiving* filter is set to *No* by default. This means, only employees that are not selected for archiving are displayed. Choose the arrow icon (▼) to expand the header and see all filter options. If you want to see employees that are selected for archiving choose *Yes* from the dropdown and *Go*. To see both, select the empty field from the dropdown.

This is how you can select an employee for archiving:

#### Prerequisite:

The employee doesn't have a user assigned.

#### Procedure

1. Open the *Maintain Employees* app and select the respective employee from the list.
2. Choose *Edit*.
3. On the *General Information* tab, in the *Validity Period* section, mark the *Selected for Archiving* checkbox.
4. Choose *Save*.

Alternatively, choose the *Select for Archiving* button.

You've selected the employee for archiving, you're now able to continue in your Information Lifecycle Management (ILM) apps.

## 6.3.1.11 Extensibility

[Custom Logic \[page 2626\]](#)

[Custom Logic \(Deprecated\) \[page 2633\]](#)

[Configure Predefined Custom Fields \[page 2638\]](#)

### 6.3.1.11.1 Custom Logic

With this app, you can implement BAdIs that have been released by SAP partners.

#### ⚠ Caution

The management of data in an extension scenario deviates from the management of data in the standard scenarios. You are responsible for ensuring that the data used in an extension scenario is managed in accordance with any applicable legal requirements or business needs, such as data protection legislation or data life cycle requirements. The extensibility framework should not be used for the processing of personal data if this processing falls under any applicable data protection legislation.

## Key Features

You can use this app to do the following:

- Create your own implementation
  - Specify filter conditions to define when an implementation is executed
- Publish your implementation to your test system
- Write custom code by using ABAP for key users
  - Test your code with predefined test variants, for example
    - Compare your code with older code versions
- Edit filter conditions of existing implementations
- Delete implementations

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-EXT-BL.

### Related Information

[Creating Implementations \[page 2627\]](#)

[Creating Custom Code \[page 2628\]](#)

### 6.3.1.11.1.1 Creating Implementations

Get an overview of how to create and publish your implementations.

### Process Steps

This is a generic description about how to create implementations. For specific documentation, refer to the individual extension point documentation in the *Custom Logic* app.

1. To create a new implementation, open the *Custom Logic* app and click *Create*.
2. Select the extension point (BAdI) that you want to implement. Perform a free text search by using the search field. The extension point descriptions, IDs and documentation will be searched. To view the documentation of an extension point, click *View Documentation*. When you have selected an extension point, click *Step 2*.
3. Depending on the extension point you've selected, you can enter the implementation attributes next. Open the *Edit Filter Condition* dialog by clicking *Add*.
4. All BAdI filter conditions added in the dialog are connected with AND. Click *Save* to add the filter condition to the list of filters. You can add more rows to the filter list by opening the dialog again. The filter list rows are connected with OR. Click *Step 3* once you've maintained the filters.
5. Enter the implementation description and the ID. The description must start with a non-whitespace character. The ID starts with a fixed prefix and its total length must not exceed 30 characters. Click *Review* once you've maintained valid values.
6. Review the data you entered. If you want to make changes, click *Edit* in the respective section. Once you're finished, click *Create*. The creation process starts in the background and you're taken to the implementation detail page.
7. Once the implementation is created, the status will change from *Creating* to *Created*. You can now edit the implementation attributes if your implementation has any, or publish your implementation to start writing code. After publishing, click *Open Code Editor* to write custom code.

## 6.3.1.11.1.2 Creating Custom Code

With the *ABAP Cloud Editor*, you can write your own code using ABAP for key users.

The editor is divided into three tabs that focus on different aspects of dealing with code: *Develop*, *Test*, and *Compare*.

### Develop

On the *Develop* tab, you can write, save, and publish ABAP code. Saving means writing a draft version of the code that only you can see. Once you publish your code, it'll be available in the system and visible for all users. Depending on the method you're implementing, there is a button *Show Sample Code*. Click this button to open a dialog that shows the sample code provided by SAP. Click *Expand* to increase the vertical space for the code.

The method signature on the right provides information about the method's parameters.

Put the cursor on a class, interface, or parameter to view the documentation displayed in the element information at the bottom of the page.

In edit mode, you can add prebuilt code blocks by selecting *Insert Code Snippet*. For more information, see [Inserting Code Snippets \[page 2629\]](#).

You can use keyboard shortcuts while working with the ABAP Cloud editor. You can view a list of all possible shortcuts by clicking *Show Keyboard Shortcuts*.

#### ⓘ Note

Do not use the `return` statement in your code because it can affect the framework logic. It can result in missing custom logic trace entries or unexpected code behavior.

### Test

On the *Test* tab, you can test your code with different parameter values. Use the parameter kind dropdown to navigate to **importing**, **exporting**, **changing**, and **returning** parameters, depending on what the method offers. Use the input fields in the table rows to enter parameter values. For table parameters, click *Add* to add a new table row or *Delete* to delete a table row.

On the right, you can choose what code version will be tested: either *Draft* or *Latest Published*. Click *Test* to run the test. If the test was successful, a toast will appear. You can then view the outcome in the different parameter tables.

If you want to save the test data you entered, click *Save As*, enter a test variant name, and click *Save*. If you have already selected an existing test variant, just click *Save*. Use *Manage* to delete saved test variants.

## Compare

On the [Compare](#) tab, you can compare different code versions. If you have a draft, it can be selected on the left. This is useful if you want to copy parts from older code versions to a new draft, for example. On the [Compare](#) tab, you can only save code but not publish it. Use the [Develop](#) tab for all code editing features.

### ⓘ Note

Please make sure that your draft logic is syntactically correct and tested. Otherwise, the draft logic might be lost when your system is upgraded to another release.

### 6.3.1.11.1.2.1 Inserting Code Snippets

Find out how to insert predefined or custom code snippets into your code.

Code snippets are small blocks of source code for recurring coding tasks such as calling a method of a custom reuse library. Some code snippets offer parameters for filling in placeholders in the code. There are two kinds of code snippets: Predefined code snippets and custom code snippets. The predefined code snippets are provided by SAP, while the custom code snippets depend on the custom entities you have created in your system.

#### Predefined Code Snippets

Predefined code snippets are calls to the custom business objects Write API. These code snippets will only be shown if the method you are implementing has the Write API as a parameter.

##### Procedure

1. Select a code snippet from the list.
2. Fill in the placeholders by maintaining the parameters on the [Details](#) page.

### ⓘ Note

The parameters are not mandatory. You can also insert the code snippet with placeholders, but then syntax errors will appear.

3. Select [Insert Code Snippet](#) to insert the code snippet into your code at the last cursor position.

#### Custom Code Snippets

Custom code snippets are calls to the methods of your custom reuse libraries. The parameters of the method call are prefilled with sample data.

### 6.3.1.11.1.3 Statements in ABAP for Key Users

This topic provides you with an overview of ABAP keywords and ABAP system fields that are available with the ABAP version for key users.

You can search the tables for specific ABAP keywords and system fields that are available with the ABAP version for key users. If you need more information and examples about how to use the ABAP version for key users, you can refer to our tutorial blog linked under *Related Information*.

ABAP for Key Users - ABAP System Fields

ABAP System Field	Restriction
sy-dbcnt	read-only access
sy-fdpos	read-only access
sy-index	read-only access
sy-subrc	read-only access
sy-tabix	read-only access

ABAP for Key Users - ABAP Keywords

ABAP Keyword	Subordinate Keyword
APPEND	
ASSIGN	
CASE	
CATCH	
CHECK	
CLEANUP	
CLEAR	
COLLECT	
CONCATENATE	
CONDENSE	
CONSTANTS	
CONTINUE	
CONVERT	Restricted to following keywords: CONVERT-DATE-TIME CONVERT-TEXT CONVERT-TIME-STAMP
DATA	
DELETE	Restricted to following keywords: DELETE-ITAB
DO	

ABAP Keyword	Subordinate Keyword
ELSE	
ELSEIF	
ENDCASE	
ENDDO	
ENDIF	
ENDLOOP	
ENDSELECT	
ENDTRY	
END WHILE	
EXIT	
EXPORT	<b>Restricted to following keywords:</b> EXPORT-DATA-BUFFER EXPORT-ITAB
FIELD-SYMBOLS	
FIND	
GET	<b>Restricted to following keywords:</b> GET-BIT GET-TIME-STAMP
IF	
IMPORT	<b>Restricted to following keywords:</b> IMPORT-DATA-BUFFER IMPORT-ITAB
INSERT	<b>Restricted to following keywords:</b> INSERT-ITAB
LOOP	<b>Restricted to following keywords:</b> LOOP-ITAB
MESSAGE	<b>Restricted to following keywords:</b> MESSAGE-INTO
MODIFY	<b>Restricted to following keywords:</b> MODIFY-ITAB
MOVE-CORRESPONDING	
OVERLAY	

ABAP Keyword	Subordinate Keyword
RAISE	<b>Restricted to following keywords:</b> RAISE-EXCEPTION
READ	<b>Restricted to following keywords:</b> READ-ITAB
REPLACE	
RESUME	
RETRY	
RETURN	
SELECT	
SET	<b>Restricted to following keywords:</b> SET-ASSOCIATION SET-BIT
SHIFT	
SORT	<b>Restricted to following keywords:</b> SORT-ITAB
SPLIT	
TRANSLATE	
TRY	
TYPES	
UNASSIGN	
WHEN	
WHILE	

## Related Information

[SAP S/4HANA Extensibility Tutorial](#)

### 6.3.1.11.1.4 Keyboard Shortcuts in Web-Based Editor

You can use keyboard shortcuts while working in the web-based editor.

The following key combinations are provided in the web-based editor:

Key Combination	What Is It Used for?
<code>Ctrl</code> + <code>Space</code>	To turn on code completion
<code>Ctrl</code> + <code>/</code>	To uncomment or comment out a code block
<code>Ctrl</code> + <code>#</code>	
<code>Shift</code> + <code>F1</code>	To use pretty printer

### 6.3.1.11.2 Custom Logic (Deprecated)

With this app you can write your own implementations to extend SAP partner code.

#### ⚠ Caution

The management of data in an extension scenario deviates from the management of data in the standard scenarios. You are responsible for ensuring that the data used in an extension scenario is managed in accordance with any applicable legal requirements or business needs, such as data protection legislation or data life cycle requirements. Please note that the extensibility framework is currently not integrated in the privacy-by-default functionality. Therefore, the extensibility framework should not be used for the processing of personal data if this processing falls under any applicable data protection legislation.

### Key Features

You can use this app to:

- Create your own implementation by using ABAP for key users
  - Implement your custom logic
  - Test your custom logic, for example with predefined test variants
  - Create filter conditions to define when an implementation is executed
- Publish implementation to your test system
- Edit implementation descriptions that have already been published
- Delete implementations

### Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-EXT-BL.

### 6.3.1.11.2.1 Creating Custom Logic

Get an overview of how to create custom logic, and how to publish your custom logic.

#### Business Benefit

This app enables you to create and maintain custom logic for business add-ins (BAdIs) that have been released by SAP partners.

#### Process Steps

This is a generic description about how to implement custom logic.

1. To find a Cloud BAdI, open the *Custom Logic* app.
2. Click **+** (*Create*). Select a BAdI Description of the delivered BAdI you want to implement. Add an implementation description and an implementation ID.
3. Choose *Create*.
4. Now you can modify and test the draft logic using ABAP for key users in a web-based editor. You can click on *Example* to see a sample implementation of the BAdI. Choose the fields for testing on the *Available Fields* tab, enter the test data and click *Test* to check if your custom logic is working as expected.

#### ⓘ Note

Please make sure that any of your draft logic is syntactically correct and tested. Otherwise, the draft logic might be lost when your system is upgraded to another Cloud release.

5. To activate your BAdI implementation, choose *Publish*.

#### Related Information

[Statements in ABAP for Key Users \[page 2635\]](#)

[Keyboard Shortcuts in Web-Based Editor \[page 2637\]](#)

[Extensibility \[page 2626\]](#)

## 6.3.1.11.2.2 Statements in ABAP for Key Users

This topic provides you with an overview of ABAP keywords and ABAP system fields that are available with the ABAP version for key users.

You can search the tables for specific ABAP keywords and system fields that are available with the ABAP version for key users. If you need more information and examples about how to use the ABAP version for key users, you can refer to our tutorial blog linked under *Related Information*.

ABAP for Key Users - ABAP System Fields

ABAP System Field	Restriction
sy-dbcnt	read-only access
sy-fdpos	read-only access
sy-index	read-only access
sy-subrc	read-only access
sy-tabix	read-only access

ABAP for Key Users - ABAP Keywords

ABAP Keyword	Subordinate Keyword
APPEND	
ASSIGN	
CASE	
CATCH	
CHECK	
CLEANUP	
CLEAR	
COLLECT	
CONCATENATE	
CONDENSE	
CONSTANTS	
CONTINUE	
CONVERT	Restricted to following keywords: CONVERT-DATE-TIME CONVERT-TEXT CONVERT-TIME-STAMP
DATA	
DELETE	Restricted to following keywords: DELETE-ITAB
DO	

ABAP Keyword	Subordinate Keyword
ELSE	
ELSEIF	
ENDCASE	
ENDDO	
ENDIF	
ENDLOOP	
ENDSELECT	
ENDTRY	
END WHILE	
EXIT	
EXPORT	<b>Restricted to following keywords:</b> EXPORT-DATA-BUFFER EXPORT-ITAB
FIELD-SYMBOLS	
FIND	
GET	<b>Restricted to following keywords:</b> GET-BIT GET-TIME-STAMP
IF	
IMPORT	<b>Restricted to following keywords:</b> IMPORT-DATA-BUFFER IMPORT-ITAB
INSERT	<b>Restricted to following keywords:</b> INSERT-ITAB
LOOP	<b>Restricted to following keywords:</b> LOOP-ITAB
MESSAGE	<b>Restricted to following keywords:</b> MESSAGE-INTO
MODIFY	<b>Restricted to following keywords:</b> MODIFY-ITAB
MOVE-CORRESPONDING	
OVERLAY	

ABAP Keyword	Subordinate Keyword
RAISE	<b>Restricted to following keywords:</b> RAISE-EXCEPTION
READ	<b>Restricted to following keywords:</b> READ-ITAB
REPLACE	
RESUME	
RETRY	
RETURN	
SELECT	
SET	<b>Restricted to following keywords:</b> SET-ASSOCIATION SET-BIT
SHIFT	
SORT	<b>Restricted to following keywords:</b> SORT-ITAB
SPLIT	
TRANSLATE	
TRY	
TYPES	
UNASSIGN	
WHEN	
WHILE	

## Related Information

[SAP S/4HANA Extensibility Tutorial](#)

### 6.3.1.11.2.3 Keyboard Shortcuts in Web-Based Editor

You can use keyboard shortcuts while working in the web-based editor.

The following key combinations are provided in the web-based editor:

Key Combination	What Is It Used for?
<code>Ctrl</code> + <code>Space</code>	To turn on code completion
<code>Ctrl</code> + <code>/</code>	To uncomment or comment out a code block
<code>Ctrl</code> + <code>#</code>	
<code>Shift</code> + <code>F1</code>	To use pretty printer

### 6.3.1.11.3 Configure Predefined Custom Fields

You, as a system administrator can use this app to configure your predefined custom fields to customize applications and their UIs.

Different companies and business processes require different document layouts or application functionalities. To address these specific needs, you can adapt predelivered fields accordingly. This app offers the option to configure predelivered custom fields in your apps to your needs.

### Key Features

You can use this app to:

- Configure a predefined field
- Provide UI texts in other languages
- Enable field usage
- Edit fields that have already been published
- Discard changes you have done to a field that was already published, and thus reset all modifications to get back to the state of the last published version of a field with one click
- Delete new fields or fields that have already been published

#### ⓘ Note

The provided functionality of predefined custom fields does not provide the technical capabilities to support the collection, processing, and storage of personal data.

### Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-EXT-FLD.

### Related Information

[Configuring Predefined Custom Fields \[page 2639\]](#)

### 6.3.1.11.3.1 Configuring Predefined Custom Fields

Get an overview of how to configure predefined custom fields, and how to publish your configured custom fields.

### Context

This process enables you to configure custom fields that were predefined by your application provider. They can be used to enhance applications which contain APIs and data sources that have been released by your software provider to configure custom fields.

### Procedure

1. Open the [Configure Predefined Custom Fields](#) app. You can open the app either directly from the SAP Fiori launchpad, or from an extensible application by clicking [+ \(Add Custom Field\)](#) in the UI adaptation mode of an application.

#### Note

To use new fields from a predefined custom field in an SAP Fiori app, you have to restart the app and the UI adaptation mode. For more information about how to adapt SAP Fiori UIs during runtime, see [Adapting SAP Fiori UIs at Runtime](#)

2. Create a new predefined custom field configuration in the business process where it is needed. To do so, select a business object, a business object node, and the technical type of the configuration. Complete this step by selecting the basic properties of this type. For example, if you selected the technical type `Character`, you need to choose one of the available field lengths your application provider has predefined for you.
3. Next, you can adapt further properties of your custom field. Enter a label and a tooltip, and choose a semantic type for your field. For example, if you selected the semantic type `Email Address`, you will be able to open your email client directly from the link.

### Caution

Once created, the semantic type of the custom field can't be changed anymore.

You can translate your field label and your field tooltip. To do so, select *Translation* in your custom field. Now, you can proceed to making your custom field available in other languages.

### Note

If your field is of type `List`, you can also translate the individual list value descriptions that belong to that field.

4. Finally, you can decide whether you want to create and publish your configured field directly, or if you want to create and edit your custom field. By selecting *Create and Edit*, you will be able to define where and how your configured field should appear in the app before publishing it.
5. You can decide which code values to enable for your custom field by using the toggles displayed in the table. Also, you can choose whether you want to make your new fields visible to the UI and app services. After you have finished editing your custom field, select *Publish*.

### Caution

Once published, you won't be able to delete the code values from the type `List` anymore. However, you can still enable or disable your code values.

### Note

If a change is made to the custom field after publishing, the status of the custom field will change to *Revised*. The changes in your custom field will only be visible in the system after selecting *Publish* once more.

You can only *Discard Changes* if the revised custom field hasn't been published yet. After publishing, you can still enable or disable specific code values in your custom field as described in step 5.

If a custom field is published or deleted, the extended application has to restart. This may result in a prolonged loading time of the extended application when it is launched for the first time.

### Caution

When you publish a field with a certain length, you can't decrease the field length later on.

Your configured custom fields will now appear in your applications and their UIs.

## Related Information

[Configure Predefined Custom Fields \[page 2638\]](#)

## 6.3.1.12 Factory Calendar

You can create and maintain holiday and factory calendars using the following apps:

- [Maintain Holidays \[page 2641\]](#)
- [Maintain Holiday Calendars \[page 2643\]](#)
- [Maintain Factory Calendars \[page 2645\]](#)

### 6.3.1.12.1 Maintain Holidays

You can use this app to display and maintain holidays according to your location, industry, or other requirements.

#### Key Features

You can use this app to:

- display existing holidays.
- edit existing holidays.
- create new holidays.
- delete holidays.

To maintain holidays, log on to the SAP Fiori launchpad in the *SAP Business Technology Platform (SAP BTP)* system and open the [Maintain Holidays](#) app.

#### ⓘ Note

To start the Fiori apps, you must have the SAP\_BR\_BPC\_EXPERT business role containing the SAP\_CA\_BC\_IC\_LND\_CAL\_CA1\_PC business catalog assigned.

#### ⚠ Caution

The following HANA SQL functions are currently not supported:

- ADD\_WORKINGDAYS
- WORKDAYS\_BETWEEN

## Displaying Existing Holidays

If you want to display existing holidays, choose one of the following options:

- Enter your search criteria in the header section and press *Go* to display a specific holiday.

- Choose a status from the *Editing Status* drop-down menu and press *Go* to display a list of holidays.

Click a holiday from the list. The holiday details screen opens.

## Creating a New Holiday

You can create a new holiday. To do so, proceed as follows:

1. Press *Create*. The *Create* dialog box opens.
2. Enter a unique holiday ID and choose one of the following holiday types from the drop-down list:
  - Fixed date: make entries in the *Month* and *Day* fields.
  - Floating dates: press *Create* and choose a date from the drop-down list.
  - Distance to easter: make an entry for the *Distance to Easter Sunday* field.
  - Fixed day from date: make entries in the *Month*, *Day* and *Weekday* fields.
3. Fill all mandatory fields and press *Create*.

## Editing Existing Holidays

You can edit an existing holiday. To do so, proceed as follows:

1. Press *Go* to display the list of existing holidays.
2. Click a holiday from the list. The holiday details screen opens.
3. Click *Edit* and make your changes in the relevant sections.
4. Press *Save* to save your entries.

## Deleting Holidays

You can delete a holiday. To do so, proceed as follows:

1. Press *Go* to display the list of existing holidays.
2. Choose a holiday from the list by checking the checkbox next to the holiday calendar ID.
3. Press *Delete*.
4. In the *Delete* dialog box, press *Delete*.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-ASF-CAL.

### 6.3.1.12.2 Maintain Holiday Calendars

You can use this app to display and maintain holiday calendars according to your location, industry, or other requirements.

#### Key Features

You can use this app to:

- display existing holiday calendars.
- edit existing holiday calendars.
- create new holiday calendars.
- delete holiday calendars.

To maintain holiday calendars, log on to the SAP Fiori launchpad in the *SAP Business Technology Platform (SAP BTP)* system and open the [Maintain Holiday Calendars](#) app.

#### ⓘ Note

To start the Fiori apps, you must have the SAP\_BR\_BPC\_EXPERT business role containing the SAP\_CA\_BC\_IC\_LND\_CAL\_CA1\_PC business catalog assigned.

#### ⚠ Caution

The following HANA SQL functions are currently not supported:

- ADD\_WORKINGDAYS
- WORKDAYS\_BETWEEN

## Displaying Existing Holiday Calendars

If you want to display existing holiday calendars, choose one of the following options:

- Enter your search criteria in the header section and press [Go](#) to display a specific calendar.
- Choose a status from the [Editing Status](#) drop-down menu and press [Go](#) to display a list of calendars.

Click a holiday calendar from the list to find more information, such as:

- Details

- Assignments
- Holiday Calendar Texts

## Creating a New Holiday Calendar

You can create a new holiday calendar. To do so, proceed as follows:

1. Press *Create*. The *Create* dialog box opens.
2. Enter a unique holiday calendar ID (*Holiday. Cal. ID*) and press *Continue*.
3. Enter values for the *Valid from* and *Valid to* fields.
4. Add an assignment and a holiday calendar text. To do so, proceed as follows:
  1. To add an assignment in the *Assignment* section, press *Create*. Choose a holiday from the drop-down list and press *Continue*. Enter a *Valid To* date and press *Continue*. Enter a *Valid From* date and press *Continue*. Press *Apply*.
  2. To add a text in the *Holiday Calendar Texts* section, press *Create*. Choose a language key from the drop-down list and press *Continue*. Enter a description in the *Description Text* field and press *Apply*.
5. Press *Save*.

## Editing Existing Holiday Calendars

You can edit an existing holiday calendar. To do so, proceed as follows:

1. Press *Go* to display the list of existing holiday calendars.
2. Click a holiday calendar from the list. The holiday calendar overview screen opens and the *Details*, *Assignments*, and *Holiday Calendar Texts* sections are displayed.
3. Click *Edit* and make your changes in the relevant sections.
4. Press *Save* to save your entries.

## Deleting a Holiday Calendar

You can delete a holiday calendar. To do so, proceed as follows:

1. Press *Go* to display the list of existing holiday calendars.
2. Choose a holiday calendar from the list by checking the checkbox next to the holiday calendar ID.
3. Press *Delete*.
4. In the *Delete* dialog box, press *Delete*.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-ASF-CAL.

### 6.3.1.12.3 Maintain Factory Calendars

You can use this app to display and maintain factory calendars according to your location, industry, or other requirements.

#### Key Features

You can use this app to:

- display existing factory calendars.
- edit existing factory calendars.
- create new factory calendars.
- delete factory calendars.

To maintain factory calendars, log on to the SAP Fiori launchpad in the *SAP Business Technology Platform (SAP BTP)* system and open the *Maintain Factory Calendars* app.

#### ⓘ Note

The business catalog SAP\_CA\_BC\_IC\_LND\_CAL\_CA1\_PC - Factory Calendar - Configuration needs to be assigned to your user to access and use the app.

The business catalog is included in the business role template SAP\_BR\_BPC\_EXPERT.

#### ⚠ Caution

The following HANA SQL functions are currently not supported:

- ADD\_WORKINGDAYS
- WORKDAYS\_BETWEEN

## Displaying Existing Factory Calendars

If you want to display existing factory calendars, choose one of the following options:

- Enter your search criteria in the header section and press *Go* to display a specific calendar.
- Choose a status from the *Editing Status* drop-down menu and press *Go* to display a list of calendars.

Click a factory calendar from the list to find more information, such as:

- Details
- Exception Rules
- Factory Calendar Texts

## Creating a New Factory Calendar

You can create a new factory calendar. To do so, proceed as follows:

1. Press *Create*. The *Create* dialog box opens.
2. Enter a unique factory calendar ID (*Fac. Cal. ID*) and press *Continue*.
3. Enter values for the *Valid from* and *Valid to* fields.
4. Press *Save*.

## Editing Existing Factory Calendars

You can edit an existing factory calendar. To do so, proceed as follows:

1. Press *Go* to display the list of existing factory calendars.
2. Click a factory calendar from the list. The factory calendar overview screen opens and the *Details*, *Exception Rules*, and *Factory Calendar Texts* sections are displayed.
3. Click *Edit*.
  1. To add a factory exception rule in the *Exception Rule* section, press *Create*. Choose a language key from the drop-down list and press *Continue*. Enter a description in the *Description Text* field and press *Apply*.
  2. To add a new text in the *Factory Calendar Texts* section, press *Create*. Choose a language key from the drop-down list and press *Continue*. Enter a description in the *Description Text* field and press *Apply*.
4. Press *Save* to save your entries.

## Deleting a Factory Calendar

You can delete a factory calendar. To do so, proceed as follows:

1. Press *Go* to display the list of existing factory calendars.
2. Choose a factory calendar from the list by checking the checkbox next to the factory calendar ID.
3. Press *Delete*.

4. In the *Delete* dialog box, press *Delete*.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-ASF-CAL.

### 6.3.1.13 i18N Services

[Language Configuration \[page 2647\]](#)

#### 6.3.1.13.1 Language Configuration

With this app you can configure the languages within a system.

### Key Features

The available language set is divided into two parts, active languages and inactive languages:

- Active languages  
This list defines those languages, which are currently available to be selected by the users.
- Inactive languages  
This list defines the set of languages, which exist in the system but are currently not offered to the users.

### Activities

If the list is displayed empty after starting the app, click button *Go* to refresh the view.

The first tab shows all currently active languages. You can remove one or more languages from this list by selecting them and then clicking the *inactivate* button.

The second tab shows all currently inactive languages. You can add one or more languages from this list to the list of active languages by selecting them and then clicking the [activate](#) button.

On both tabs you can use the fulltext [Search](#) function to find one or more languages in the list.

On both tabs you can filter the displayed languages by language code. To refresh the tabs display click button [Go](#).

## Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-I18.

### 6.3.1.14 Identity and Access Management

Identity and Access Management (IAM) enables you to control user access to apps and specify what business users can do and see in the apps.

The main elements of IAM are business catalogs, business roles, and business users. The IAM apps secure the access to your solution based on these elements. Please see [Business Roles](#), [Business Catalogs](#), and [Restrictions](#) in the [Related Information](#) section for a more detailed description of how these elements interrelate.

Access to business apps is controlled by a role-based authorization management. That means you assign business roles to users and these business roles provide access to certain business tasks.

You can also assign business users to business roles in the **Maintain Business Roles** app. The main purpose of the app though is to create and adapt business roles. You create business roles by combining pre-defined business catalogs that contain the actual authorizations that allow users to access apps for a specific business area. If necessary, you can change the restrictions for the access categories value help, read, and write on field level. Once you have created a business role, you can assign it to multiple business users who perform similar business tasks in the **Maintain Business Users** app. Business users are all persons that need access to your solution, such as employees or contractors.

As opposed to business users, technical users aren't persons, but rather services that are used to automate technical tasks in the system, for example, a communication user. The **Display Technical Users** app shows you what technical users are available in your solution.

### Note

We recommend that you use the [Manage Business Role Changes After Upgrade](#) app as a starting point to carry out regular release activities. For more information, see [Manage Business Role Changes After Upgrade](#) in the [Related Information](#) section.

### Note

To find out which business roles grant access to which apps, you can use the [IAM Information System](#) app. On the [Business Role - Application](#) tab, you can display all apps per business role as well as the corresponding business catalog IDs. If you select [Application](#) in the [Main Entity](#) dropdown list, you can search for apps and see which business roles and business catalogs provide access to them. For more information, see [IAM Information System](#) in the [Related Information](#) section.

If you don't have access to the [IAM Information System](#) app, you can still find out which business catalogs grant access to which apps using the SAP Fiori Apps Reference Library. For more information, see the [Related Information](#) section.

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## Access to the Apps

To have access to these apps, your user needs to be assigned to a business role which contains one of the following business catalogs:

- *Identity and Access Management* (SAP\_CORE\_BC\_IAM)
- *User Management* (SAP\_CORE\_BC\_IAM\_UM)
- *Role Management* (SAP\_CORE\_BC\_IAM\_RM)
- *Role Assignment* (SAP\_CORE\_BC\_IAM\_RA)

For more information, see sections [Maintain Business Users](#) and [Business Catalogs for Identity and Access Management Apps](#) linked below or ask your administrator.

## Related Information

[Maintain Business Users \[page 2655\]](#)

[Display Technical Users \[page 2692\]](#)

[Business Catalogs \[page 2699\]](#)

[IAM Information System \[page 2701\]](#)

[Business Role Templates \[page 2702\]](#)

[Business Catalogs for Identity and Access Management Apps \[page 2706\]](#)  
[Maintain Deleted Business Users \[page 2662\]](#)  
[Manage Business Role Changes After Upgrade \[page 2685\]](#)  
[IAM Key Figures \[page 2703\]](#)  
[Display Authorization Trace \[page 2704\]](#)  
[Display Restriction Types \[page 2685\]](#)  
[Maintain Business Roles \[page 2666\]](#)  
[Business Roles, Business Catalogs and Restrictions \[page 2651\]](#)  
[Manage Business Role Changes After Upgrade \[page 2685\]](#)  
[IAM Information System \[page 2701\]](#)  
[SAP Fiori Apps Reference Library - User Guide](#)

### 6.3.1.14.1 Users

There are different categories of users for different purposes and with different capabilities. Business users represent end users performing day-to-day business tasks. They constitute personal data and their lifecycle is governed by retention and deletion policies.

Technical users are used in the background for system tasks such as inter-system communication, printing, and support.

## Business Users

### Relation of Employees and Business Users

Employees are metadata-records containing personal and employment-related data. Employees do not have access to the system directly, but only via corresponding business users.

A business user corresponds to a real-life person who works with the system interactively. Business users are created to allow a person access to the system. You can maintain business users in the [Maintain Business Users](#) app. Authentication and logon for business users happens at the identity provider.

### Business User Authorizations and Monitoring

You can assign business users to business roles in the [Maintain Business Roles](#) app. The main purpose of the app though is to create and adapt business roles.

Information about currently assigned business roles can be found in the [IAM Information System](#) app and the [IAM Key Figures](#) app.

### Business User Lifecycle

Business users that haven't been used for a certain time can be automatically locked.

Retention and ultimate deletion of blocked business users is governed by the Information Lifecycle Management (ILM) component.

### Note

Please note that ILM is not available in SAP BTP ABAP Environment.

## Technical Users

Technical users correspond to a local or remote process which is typically part of the cloud management process (such as system provisioning, support) or intrinsic system processes. There are technical users that belong to the software or service provider and there are technical users that belong to the customer.

You can get an overview of all technical users in the [Display Technical Users](#) app.

### Communication Users

The communication user is a technical user type aimed at technical communication. A communication user corresponds to a remote system that connects to your own system. You can define communication users in the [Maintain Communication Users](#) app.

### Support Users

Support users are a separate type of technical user for temporary access. These users are intended for SAP employees who access the system based on a support ticket from the customer. The [Display Technical Users](#) app shows which support users were created for the system on which incident.

## Related Information

[Maintain Business Users \[page 2655\]](#)

[Maintain Business Roles \[page 2666\]](#)

[IAM Information System \[page 2701\]](#)

[IAM Key Figures \[page 2703\]](#)

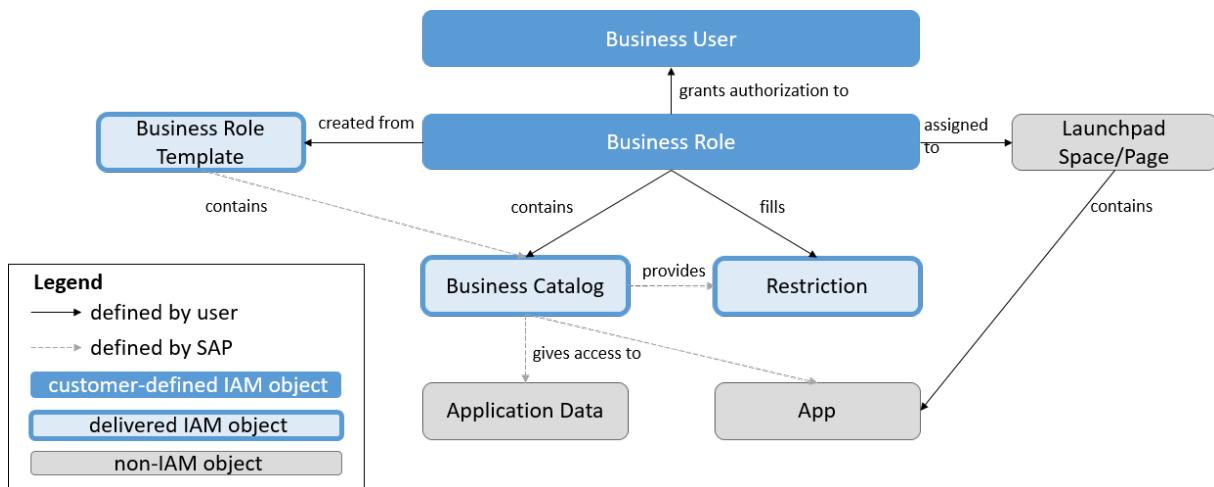
[How to Lock Unused Business Users \[page 2658\]](#)

[Display Technical Users \[page 2692\]](#)

[Maintain Communication Users \[page 2593\]](#)

## 6.3.1.14.2 Business Roles, Business Catalogs and Restrictions

SAP divides the business functionality into semantically meaningful business catalogs, representing tasks or sub-processes within a business process. These catalogs are the most finely grained units regarding structuring of work and authorization assignment. Business catalogs grant access to an app, a set of apps, or individual aspects of an app. Some business catalogs have restrictions. These restrictions give you the option of further specifying the way the user might interact with the app: they may, for example, grant write or read access.



Business catalogs are grouped into collections called business roles. A business role generally contains multiple business catalogs and corresponds to a set of authorizations required to perform the tasks of a particular job description, for example, a warehouse clerk. On the role level, restriction values of the contained business catalogs are defined. A business catalog might be contained in different business roles and might have different restriction values assigned in these different business roles.

Launchpad spaces and pages structure the launchpad layout. You can assign launchpad spaces to business roles so that the launchpad space is visible to all users with a given business role.

Business roles are assigned to business users. Multiple business roles might be assigned to a single business user. And a business role might of course be assigned to different business users.

To facilitate the process of business role creation, SAP delivers business role templates. These templates are collections of business catalogs suitable for a particular user persona. You can create a business role by simply copying the business role template and specifying restriction values.

Please note that it is not recommended to use copies of business role templates for productive use.

### ① Note

Please note that it is not recommended to use copies of business role templates for productive use.

## Implementation Considerations

During the implementation phase of an IAM concept, the following considerations might be useful. After implementation, mechanisms for monitoring and analysis become more important.

The lifecycle of IAM elements is influenced by system upgrades. For more information, see [Manage Changes and Deprecation after Upgrades](#) in the [Related Information](#) section.

For more information about monitoring of the IAM configuration, see [Monitoring and Analysis](#) in the [Related Information](#) section.

### Business Role Implementation

If you want to model business roles sharing common properties, you can make use of the inheritance mechanism of leading and derived business roles.

In the context of inheritance, you can maintain the intended level on which a field should be filled in a business role using the field maintenance status.

#### Transport

Changes to business roles should not be carried out in production systems directly.

We recommend that you define business roles in a development system and transport them into the production system instead of making local changes.

### Related Information

[Monitoring and Analysis \[page 2653\]](#)

[Manage Changes and Deprecations After Upgrade \[page 2654\]](#)

[How to Define Authorizations Based on Restrictions \[page 2672\]](#)

[How to Create Leading and Derived Business Roles \[page 2677\]](#)

### 6.3.1.14.3 Monitoring and Analysis

Monitoring and analysis tasks are part of the daily work as administrator.

Common activities in this context include the following:

- Find business users with unnecessarily high privileges (for example, too many admin users)
- Find business users with unnecessary or critical authorization combinations (the result of privilege creep)
- Find business roles granting access to data too generously (that is, having unmaintained restrictions)
- Analyze the root cause for authorization errors

You can use the [IAM Information System](#) app to monitor the relationships between business users, business roles, business catalogs, and so on.

The [IAM Key Figures](#) app enables you to monitor KPIs for your authorization system. It shows the number of locked business users, for example. Also, the number of unrestricted business roles is shown. Thresholds can be defined to visualize critical deviations from the desired state.

For the analysis of issues with privileges you can use the [Display Authorization Trace](#) app.

An overview of the technical users in the system – both communication users and SAP support users – is provided by the [Display Technical Users](#) app. The [Maintain User Sessions](#) app enables you to analyze sessions containing a lock and to find the associated business user.

### Related Information

[IAM Information System \[page 2701\]](#)

[IAM Key Figures \[page 2703\]](#)

[Display Authorization Trace \[page 2704\]](#)

[Maintain User Sessions \[page 2779\]](#)

### 6.3.1.14.4 Manage Changes and Deprecations After Upgrade

Your system is upgraded regularly to deliver new features and improvements.

These changes may include changes or deprecation of apps, business catalogs, and business role templates.

#### Deprecation Process for Apps

If an app is outdated or its functionality is covered by a successor app, it is deprecated. This means that even though you can still use the app, SAP no longer recommends its use, and it will be deleted from the SAP Fiori launchpad in an upcoming release. There's a period of at least six months between the initial announcement of an app's deprecation and its deletion. The deprecated app may no longer be available by default on the SAP Fiori launchpad. In this case, you can find it in the app finder until it is deleted.

The deprecation of an app may require you to recheck the business catalogs that are assigned to your business roles. For more details about this and changes to restriction types, see [Manage Business Role Changes After Upgrade](#) in the *Related Information* section.

#### Deprecation of Business Catalogs

Due to ongoing development of new features and new apps, you need to revise existing business catalogs with every system upgrade. This means that some business catalogs are deprecated and replaced by new ones, and you need to assign business roles and business users to these new business catalogs.

Rather than disappearing, deprecated business catalogs are identified as being obsolete, which allows you to identify them at a glance. For more information, see [How to Handle Deprecated Business Catalogs](#) in the *Related Information* section.

#### Business Role Changes after Upgrades

It is important for you to know how the release changes impact your productive business roles.

Business role changes often have the technical reason that new developments may require changes to existing applications, business catalogs, restriction types, and/or restriction fields. The affected business roles need to be adapted accordingly to ensure that existing apps behave consistently before and after the upgrade.

The following typical changes might occur:

- Business catalogs were deprecated, or split, or have new dependencies so new business catalogs need to be assigned.

- New apps were added to existing business catalogs.
- New restriction types were added, removed, or access categories were changed.

For more information about change activation, see *Phase-In/Phase Out Status* in the *Related Information* section.

## Related Information

[Manage Business Role Changes After Upgrade \[page 2685\]](#)

[How to Handle Deprecated Business Catalogs \[page 2700\]](#)

### 6.3.1.14.5 Maintain Business Users

You use this app to provide business users with access rights and to maintain business user settings.

Business users gain access to Fiori apps through business roles. A business role can comprise one or several business catalogs which in turn comprise several apps.

## Key Features

You can use this app to:

- Edit business user data
- Assign business roles to a user and remove existing assignments
- Update user role assignments
- Lock users so that they cannot log on to the system
- Unlock users
- Download list of users
- Set the language of the user interface for each business user

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

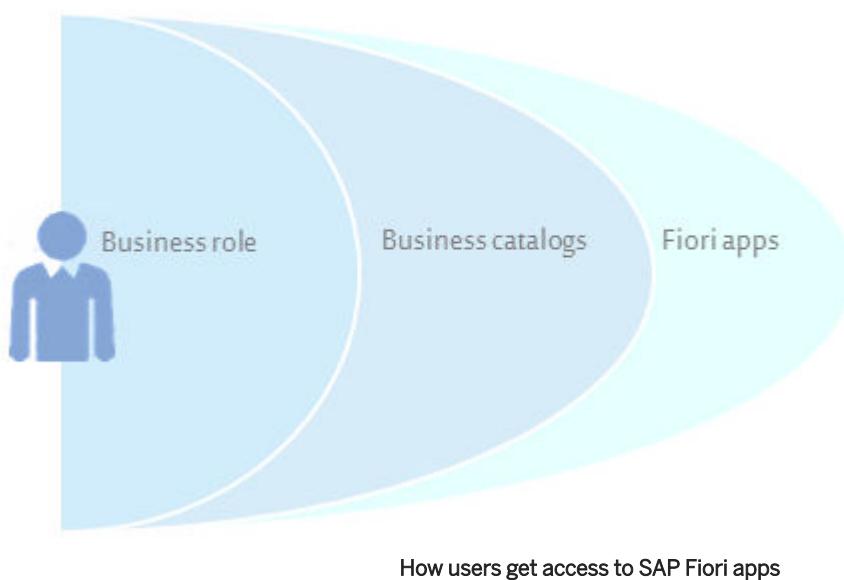
## Related Information

[Business Catalogs for Identity and Access Management Apps \[page 2706\]](#)

### 6.3.1.14.5.1 How to Maintain Business Users

#### Context

Business users are replicated from your central HR system. You can change user data (for example, the user name) and regional settings (for example, the date and time format).



#### Procedure

- **Creating New Business Users**
- To create a new business user, proceed as follows:
  - a. Choose *Create* on the main screen of the app.
  - b. Select the employee you want to create the user for, and fill in the required fields.

For *User Name*, enter a name or ID that is identical with the login name of the same user in the on-premise identity provider.

The name the user enters to log on to the SAP Fiori launchpad. The user name can only contain uppercase letters, numbers, and underscores. It must have a maximum length of 40 characters and must not start with an underscore, or with SAP.

- **Assigning Roles to Business Users**

- To grant users access to applications, you assign the respective business roles to them.
  - a. Select the required user.
  - b. Click *Add Business Roles*.
  - c. Search for the required roles and select them.
  - d. Click *Apply* and *Save*.

You can navigate from the Maintain Business Users app to the Maintain Business Roles app by choosing a business role ID in the list of assigned business roles.

The user then sees the respective app tiles in the tile catalogs.

- **Removing Role Assignments from Business Users**
- To remove a role assignment from a user, proceed as follows:
  - a. Select the business user and then the assigned business role.
  - b. Choose *Remove* above the list or *Remove Business Roles* at the bottom of the screen.
- **Updating User Role Assignments**
- To update the assignments of roles to a user individually, edit the affected business user.
- To mass update user role assignments, upload a csv file.

#### ⓘ Note

The CSV file needs to be UTF-8-encoded and must comply with the following pattern:

User Name	User ID (Optional)	E-Mail (Optional)	Global User ID (Optional)	Role ID
<User name 1>	<Business user ID 1>	<Business user's mail address 1>	<Global user ID 1>	<Business role ID 1>
<User name 2>	<Business user ID 2>	<Business user's mail address 2>	<Global user ID 2>	<Business role ID 2>
<User name 3>	<Business user ID 3>	<Business user's mail address 3>	<Global user ID 3>	<Business role ID 3>
<User name n>	<Business user ID n>	<Business user's mail address n>	<Global user ID n>	<Business role ID n>

In the app, you are provided with a csv template that you can download and use for filling in the user role assignments. When uploading the csv file, you can decide if you want to add the role assignments to a user, or if you want to overwrite them completely.

You can find the CSV template as follows: Click the *Upload* button, then click *Upload User Role Assignments* in the menu that appears. This opens a dialog box containing the *Download CSV Template* link that you can use to download the template.

The user ID is optional. Only if the user name is not unique, the user ID has to be maintained.

- **Downloading User Lists**
- To download a list of all users with an email address as csv files for upload to *SAP Cloud Identity Services - Identity Authentication*, proceed as follows:

- a. To open the file, you have to set the list separator to comma by setting the format in the regional settings of your operating system to *English (United States)*.
- b. Open the *Administration Console for SAP Cloud Identity Services – Identity Authentication*.
- c. To upload the csv file, go to  *User Management*  *Import Users* .

 **Note**

Make sure that the csv file is not opened with any program before the upload.

- Changing the Default Settings for a Business User

To change the default settings for a business user, select it for editing and adjust the values as required. If you create a new business user, the following fields in the *User Data* area and the *Regional Settings* area are automatically filled with default values. You can adjust them if necessary.

Default Values

Field	Default Value
<i>Valid From</i>	01/01/1901
<i>Valid To</i>	12/31/9999
<i>Decimal Format</i>	This default value is derived from the standard setting that is valid in the country of the workplace address. If the workplace address is not maintained, the value is derived from the business user's private address.
<i>Date Format</i>	
<i>Time Format</i>	
<i>Time Zone</i>	
<i>Language</i>	

- 

## Related Information

[Defaulting of User Attributes](#) 

## 6.3.1.14.5.2 How to Lock Unused Business Users

Lock inactive business users

## Context

You can schedule a job to automatically lock business users who haven't logged on for a certain number of days.

### Note

To schedule this job, you can either use the [Application Jobs](#) app as described below or you can add the *Schedule Lock for Unused Business Users* tile your SAP Fiori Launchpad using [Edit Homepage](#) if you want to access the repetitive scheduling screen directly.

## Procedure

1. Open the [Application Jobs](#) app.
2. Under [Job Templates](#), select [Lock Unused Business Users](#) and click [Step 2](#).
3. Define the recurrence pattern of your job and click [Step 3](#).
4. Under [Parameters](#), enter the required number of days since the last logon you want to use for the automatic lock.
5. Add the business users you want to exclude.

Optional:

Open the value help if you want to define further conditions.

Go to the [Define Conditions](#) tab. Here you can define single users or whole user areas you want to exclude from the selection. You can enter multiple users at once by using copy and paste, for example to transfer users from lists.

Please note that you can also restrict the list of users that should be processed by this job.

6. Click [Schedule](#).

## 6.3.1.14.5.3 How to Make Mass Changes to Business Users

## Context

You can use the mass change wizard to change multiple business users at once. This is helpful, for example, if you want to apply the same validity period to more than one business user.

## Procedure

1. Select the business users you want to change and click [Mass Change](#). The [Mass Change Wizard](#) appears. It has the following pattern:
2. Select the required attributes. You can make the following changes:

Area	Attributes
User Data	Select one or more attributes (such as timezone or language) you want to change in all selected business users.
Role Assignment	Add Business Roles
	Remove Business Roles

3. Apply the required attributes to your business users.
4. Review and confirm your changes.

### 6.3.1.14.6 Maintain Business User Groups

With this app you can create business user groups and assign multiple business users to them. This helps you to organize your area and easily search for all business users of a certain category (for example to assign them to business roles). Grouping also facilitates maintenance of authorizations. If you are the super administrator for all areas, you can delegate this task to administrators for the relevant areas, such as Financials. In this case, you would create a business user group for Financials.

Please note that the name of business user group is created automatically in namespace [ZCB](#).

#### Key Features

You can use this app to:

- Create business user groups and assign the required business users to them
- Reassign/remove business users to/from business user groups  
Note: You can also use drag and drop to move business users from one group to another.
- Upload business user assignments
- Add business user groups to transports
- Export business user groups to a spreadsheet (including description and number of business users)

#### Prerequisites

The following restriction types (in the following business catalogs) are required for maintaining business role groups and business user groups:

Business Catalogs	Restriction Types
SAP_CORE_BC_IAM_RM	<a href="#">Business Role</a> (S_BRL)
SAP_CORE_BC_IAM_UM	<a href="#">Business User</a> (CLASS)

Business Catalogs	Restriction Types
SAP_CORE_BC_IAM_RA	<i>Business Role</i> (S_BRL) <i>Business User</i> (CLASS)
SAP_CORE_BC_IAM_RM_DISP_PC	<i>Business Role</i> (ROLE_GROUP)
SAP_CORE_BC_IAM_UM_DISP_PC	<i>Business User</i> (CLASS)

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.6.1 How to Transport Business User Groups

#### Context

You can export business user groups from your quality system to your productive system using customizing transports.

#### Procedure

1. In the *Business User Groups* overview screen of the *Maintain Business User Groups* app, select one or more business user groups.
2. Click *Add to Transport*. A list of available customizing transports appears.

#### ⓘ Note

The default transport is always displayed as well as your own transports. If no transport exists, a confirmation dialog is shown. If you confirm, a new transport is created.

3. Select the required transport and click *OK*. The selected business user groups are then added to the transport.

4. Open the *Export Customizing Transports* app and release your request. For more information, see [Export Customizing Transports \[page 2582\]](#).
5. Import your business user groups to the target system:
  - a. Open the *Manage Software Components* app.
  - b. Select *Business Configuration* in the *Type* dropdown and click *Go*.
  - c. Select the component to which the relevant configuration was exported.
  - d. In the detail view of the component, check out the change that contains your business user group(s). For more information, see [How to Work with Branches](#).

 **Note**

If you want to delete a business user group that was previously transported or is currently included in an open transport request, the system offers you to add it directly to an available transport.

If the business user group was previously transported and there is currently no transport available, a confirmation dialog is shown. If you confirm, a new transport is created.

### 6.3.1.14.7 Maintain Deleted Business Users

With this app you can display details of deleted business users, such as the retention period, and allow or block their re-creation.

#### Key Features

You can use this app to:

- Display a list of all deleted users including deletion time, end of re-creation period and re-creation status
- Resolve the identity of deleted business users by displaying contact details such as email address or phone number
- Change the re-creation status from *Allowed* to *Not Allowed* or the other way round.
- View system deletion checks to find out if a user can be permanently deleted or if it is still used or to check why blocked user data wasn't destroyed.

 **Note**

To access the *Maintain Deleted Business Users* app, you need to assign the SAP\_CORE\_BC\_IAM\_UMD business catalog to the particular business role.

#### Supported Device Types

- Desktop

- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.7.1 How to Enable the Recreation of Blocked Business Users

Sometimes the person with whom a deleted business user was associated must access the system again.

#### Context

The system protects users in the blocking area from being recreated. The system prevents blocked business users from being recreated to prevent inconsistencies in the user data. You don't want the same business user associated with two different people. But there are cases where you want to recreate business users, which have been deleted. Perhaps a person who left your organization has come back.

##### ⓘ Note

If you have the authorization to allow the recreation of users and you are recreating a blocked user, the recreation is automatically allowed.

#### Procedure

1. In the app *Maintain Deleted Business Users*, select the business user you want to recreate.
2. Click *Allow Re-creation*.

#### Results

Use your standard processes for creating this business user again in the system. When the business user is recreated, the system removes the business user entry from the app *Maintain Deleted Business Users*.

##### ⓘ Note

The system does not transfer the address data from the app *Maintain Deleted Business Users*. Reenter the address information in your process for business user creation.

## 6.3.1.14.8 Maintain Business Role Groups

With this app you can create business role groups and assign multiple business roles to them. This helps you to organize your area and easily search for all business roles of a certain category (for example to assign business users to them). Grouping also facilitates maintenance of authorizations. If you are the super administrator for all areas, you can delegate this task to administrators for the relevant areas, such as Financials. In this case, you would create a business role group for Financials and allow to assign these business roles only to particular user groups.

Please note that the name of business role group is created automatically in namespace [ZCB](#).

### Key Features

You can use this app to:

- Create business role groups and assign the required business roles to them
- Reassign/remove business roles to/from business role groups  
Note: You can also use drag and drop to move business roles from one group to another.
- Upload business role assignments
- Add business role groups to transports
- Export business role groups to a spreadsheet (including description and number of business roles)

### Prerequisites

The following restriction types (in the following business catalogs) are required for maintaining business role groups and business user groups:

Business Catalogs	Restriction Types
SAP_CORE_BC_IAM_RM	<a href="#">Business Role</a> (S_BRL)
SAP_CORE_BC_IAM_UM	<a href="#">Business User</a> (CLASS) <a href="#">Business Role</a> (S_BRL)
SAP_CORE_BC_IAM_RA	<a href="#">Business Role</a> (S_BRL) <a href="#">Business Role User Assignment</a> (S_BRL_ASG) <a href="#">Business User</a> (CLASS)
SAP_CORE_BC_IAM_RM_DISP_PC	<a href="#">Business Role</a> (S_BRL)
SAP_CORE_BC_IAM_UM_DISP_PC	<a href="#">Business User</a> (CLASS)

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.8.1 How to Transport Business Role Groups

#### Context

You can export business role groups from your quality system to your productive system using customizing transports.

#### Procedure

1. In the *Business Role Groups* overview screen of the *Maintain Business Role Groups* app, select one or more business role groups.
2. Click *Add to Transport*. A list of available customizing transports appears.

##### ⓘ Note

The default transport is always displayed as well as your own transports. If no transport exists, a confirmation dialog is shown. If you confirm, a new transport is created.

3. Select the required transport and click *OK*. The selected business role groups are then added to the transport.
4. Open the *Export Customizing Transports* app and release your request. For more information, see [Export Customizing Transports \[page 2582\]](#).
5. Import your business role groups to the target system:
  - a. Open the *Manage Software Components* app.
  - b. Select *Business Configuration* in the *Type* dropdown and click *Go*.
  - c. Select the component to which the relevant configuration was exported.
  - d. In the detail view of the component, check out the change that contains your business role group(s). For more information, see [How to Work with Branches](#).

#### Note

If you want to delete a business role group that was previously transported or is currently included in an open transport request, the system offers you to add it directly to an available transport.

If the business role group was previously transported and there is currently no transport available, a confirmation dialog is shown. If you confirm, a new transport is created.

### 6.3.1.14.9 Maintain Business Roles

With this app you can access the maintenance UI for restrictions as well as the [Restrictions Overview](#).

With the [Maintain Business Roles](#) app, you can also define business roles by combining predefined business catalogs. You use business roles to control the access to your applications. The predefined catalogs contain the actual authorizations that allow users to access apps and allow to define instance-based restrictions where necessary. Business catalogs bundle authorizations for a specific business area. Once you have created a business role, you can assign it to multiple business users who perform similar business tasks.

#### Key Features

You can use this app to:

- Display the [Restrictions Overview](#)
- Maintain restrictions in a new UI
- Create business roles from scratch (based on selected business catalogs)
- Create business roles from a template delivered by SAP
- Download business roles from the test system and then upload them to the productive system per XML file
- Assign one or more business roles to a business user
- Copy business roles
- Delete business roles

#### Supported Device Types

- Desktop
- Tablet

#### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## Related Information

[Maintain Restrictions UI \[page 2674\]](#)

[Restrictions Overview \[page 2676\]](#)

### 6.3.1.14.9.1 How to Create a New Business Role

Get an overview of how to create a new business role from scratch based on selected business catalogs.

#### Context

You use business roles to control the access to your applications. To create a business role, you add one or multiple business catalogs to it. These predefined catalogs contain the actual authorizations that allow users to access apps and allow to define instance based restrictions where necessary. Business catalogs bundle authorizations for a specific business area. Once you have created a business role, you can assign it to multiple business users who perform similar business tasks.

#### ⓘ Note

Please note that this document describes how to create a business role without a template. For more information on how to create a business role from a template, see the [Related information](#) section.

#### Process Steps



#### Procedure

1. Select the tile of the Maintain Business Roles app on the SAP Fiori Launchpad to open the app. On the initial screen, select [New](#).
2. Add general role details, such as business role name, ID and description.
3. On the [Assigned Business Catalogs](#) tab, select [Add](#) to add the required business catalogs. Select the catalogs according to the business activities that the users with this role need to perform. Select [Apply](#).

#### ⓘ Note

Some business catalogs require additional dependent catalogs to be assigned to enable access to associated master data information (for example, for business partners or customers). These additionally required catalogs ensure access to all business objects used with the SAP Fiori apps of the main catalog. When you select the business catalogs you want to add to the business role and click

[Add](#), a list of dependent business catalogs is displayed. You can then select all the dependencies you want to add.

4. By default, the value help and read access for each business catalog is set to unrestricted and there is no write access. If you want to change these restrictions, select [Maintain Restrictions](#).
5. Maintain instance-based restrictions for all required business objects (following the requirements of your local authorization concept).
6. If required, click [Manage Launchpad Space](#) to create a new space or use an existing space. For more information, see the [Related information](#) section below.

#### ⓘ Note

The group-based home page is deprecated. It will be removed in a future version. For new systems (starting with SAP S/4HANA Cloud 2302), the group-based home page is no longer available. It will be replaced with the spaces and pages mode. Therefore, a warning message is shown. It appears when you save a business role without an assigned space. It reminds you that you need to assign a space to make the tiles visible for the users when you have already switched to the spaces and pages mode.

For more information, see SAP Note [2970113](#).

7. On the [Assigned Business Users](#) tab, you can assign the business users to your new business role. These users will receive the authorizations as defined in the business role.
8. Save the business role to activate it.

#### ⓘ Note

If you go back to the business roles overview **without saving** the business role, the business role will automatically be saved in a draft status. You can access it again and edit it from the business roles overview.

## Related Information

[Step by Step: Create a New Space and Page for a Business Role](#)

[How to Create a Business Role from a Template \[page 2668\]](#)

## 6.3.1.14.9.2 How to Create a Business Role from a Template

Get an overview of how to create a business role from a template.

## Context

Instead of creating a business role from scratch, you can also create it from a business role template.

### **ⓘ Note**

A business role template is defined by SAP to make it easier for you to find the business catalogs that might be relevant for the corresponding role in your company. Usually, the business role templates have a very broad job profile to show all options. SAP does not recommend using them in their full scope because the business catalogs might conflict with the roles in your company. Instead, adjust them to the tasks of the role in your company by choosing only the relevant business catalogs. We also recommend that you only use business role templates for testing and not for generating productive roles.

If changes to the template were included in an upgrade, the *Business Role Templates* app informs you about these changes and how they affect your business roles. For more information, see *Business Role Templates* (Related Information).

When you create a role based on a template, as a default the read and value help access are unrestricted, and write access is not granted. You can change this and define for each field to which a catalog provides access what a user is allowed to see and to edit. This allows you to shape your business roles in a very detailed way. For example, you can create two roles that comprise the same catalogs, but role 1 is restricted to work with data for the US, and role 2 is restricted to work only with data for Germany.

### **Process Steps**



### **Procedure**

1. Select the tile of the *Maintain Business Roles* app on the SAP Fiori launchpad to open the app. On the initial screen select *Create From Template*.
2. Select the required template. Define the ID and the name of the new business role. If required, go to the *Assigned Launchpad Spaces* tab, click *Add*, select *Use Predefined Space*, select the required launchpad space and click *Assign Space* (for more information, see the *Related Information* section below). Click *OK*.
3. A template already contains one or more business catalogs that the business role will be assigned to. Adjust the displayed template to your requirements, for example, change the general role details, and add or delete catalogs.
4. By default, the value help and read access for each business catalog is set to unrestricted and there is no write access. If you want to change these restrictions, select *Maintain Restrictions*.
5. Maintain instance-based restrictions for all required business objects (following the requirements of your local authorization concept).
6. On the *Assigned Business Users* tab, you can assign the business users to your new business role. These users will receive the authorizations as defined in the business role.
7. Save the business role to activate it.

### **ⓘ Note**

If you go back to the business roles overview **without saving** the business role, the business role is automatically saved in a draft status. You can access it again and edit it from the business roles overview.

## Related Information

[Business Role Templates \[page 2702\]](#)

[Step by Step: Create a New Space and Page for a Business Role](#)

### 6.3.1.14.9.3 How to Create a Business Role for the Administrator

Get an overview of how to create a business role for the administrator if you haven't used the system before.

## Context

If you as an administrator haven't used the Fiori apps before, you have to create an administrator business role first. Otherwise the app tiles are not visible on the Fiori Launchpad. With this role, you can create all other business roles according to the needs of your company.

### ⚠ Caution

The SAP\_BR\_ADMINISTRATOR role is pre-defined by SAP and only intended for the initial set-up of a system. Please create your own business roles based on the required business role template for productive use.

To create a business role for the administrator, proceed as follows:

## Process Steps



## Procedure

1. Use the initial credentials that you have received separately e.g. via email, to log in to the system. On the SAP Fiori Launchpad select the tile *Maintain Business Roles* to open the app.
2. Select *Create From Template*. The *Create Business Role from Template* window opens. In the field *Template*, search for SAP\_BR\_ADMINISTRATOR. Define the ID and the name of the new business role.
3. A predelivered template contains a number of catalogs that in this case an administrator could need to perform tasks. These catalogs are displayed on the *Assigned Business Catalogs* tab. By default for each catalog the read access is unrestricted and there is no write access. If you want to change the restrictions, for example, to allow unrestricted write access for all catalogs, proceed as follows:
  - a. Select *Maintain Restrictions*.

- b. On the new screen, select the restriction you would like to change.  
Under *Write*, select *Unrestricted*.
  - c. Select *Back to Main Page* to save the changes.
4. On the *Assigned Business Users* tab, all users that are assigned to the selected catalogs are listed. To add a business user, proceed as follows:
    - a. Select *Add*.
    - b. In the *Add Business Users* window, search for your own business user and select *Apply*.
  5. Save the business role to activate it.

**ⓘ Note**

If you go back to the business roles overview **without saving** the business role, the business role will automatically be saved in a draft status. You can access it again and edit it from the business roles overview.

You can now log out from the system and log in with your own credentials and proceed creating further business roles according to the needs of your company

### 6.3.1.14.9.4 How to Edit Active Business Roles

#### Context

The editing of an active business role takes place in a draft version of this business role. This draft version is created by the system once you enter the edit mode of the active business role. Proceed as follows to edit an active business role:

Select the tile of the Maintain Business Roles app on the SAP Fiori Launchpad to open the app. On the initial screen, select

#### Procedure

1. Select the tile of the *Maintain Business Roles* app on the SAP Fiori Launchpad to open the app.
2. On the initial screen, open the business role you need to edit by clicking on it in the *Business Roles* list. Alternatively, you can either use the *Search* or filter for the business role ID, for example.
3. To enter the edit mode, click *Edit*.

A draft version of the business role is created in which the editing takes place. The active business role remains.

4. Make the required changes to the business role.
5. Save the business role.

Once the changes are saved, they are written into the active business role and the draft version disappears. **If the dialog for editing a business role is left without saving the business role**, the active business role

as well as the draft version of the business role will be available in the business roles overview. In this case, you can continue editing the draft version later.

### 6.3.1.14.9.5 How to Define Authorizations Based on Restrictions

#### Context

The authorization concept for business roles is based on the concept of granting authorizations (as opposed to denying them). By adding business catalogs to a business role and assigning the business role to a business user, you can control which applications the business user is authorized to carry out. As a next step, you define how the customer data can be accessed. You do this by adding authorization values to the restriction fields. Restriction fields are organized in restriction types. Restriction types that contain general organizational restriction fields can be found in the top section called *General*. These restriction types contain only one restriction field. The settings you make for the single restriction types sum up to the authorization that is granted to the business role and therefore to the assigned business users.

##### ⚠ Caution

Please note that assigning multiple business roles to a business user increases the risk of overriding existing authorizations.

For example, you have created the business roles 1 and 2 that both include the business catalog A. In business role 1, you restricted the access rights for sales organization to A. In business role 2, you allowed to work with data for all sales organizations. The business user to whom you assign both roles will then have full access to the data for all sales organizations.

This overriding effect can also be caused if one restriction type is used in different assigned business roles of different business catalogs.

Using the *Maintain Business Roles* app, you have the following options for maintaining restrictions:

#### Procedure

- In the *Maintain Restrictions* UI, you can maintain restrictions for business roles. These restrictions are the basis of authorizations that are granted to the business users who are assigned to this business role. The business catalog defines which access categories are available for maintaining and for which fields restriction values can be maintained.

The following access categories are available:

- Write, Read, Value Help* (write access)
- Read, Value Help* (read access)

- [Value Help](#) (value help access)

The business role aggregates the authorizations of the assigned catalogs.

The available restriction fields represent the authorization-relevant attributes of the business objects that are used in a role. Authorization for these fields can be granted on write, read or value help level (for example for a particular sales area).

You can select or enter single values (pre-defined by SAP or customer-specific configuration) and ranges.

For more information, see [Display Restriction Types](#) (Related Information).

### Note

When you set an access category to [Restricted](#) you can define the data access for each restriction type and restriction field according to your process requirements. You can do this on [Read](#), [Write](#) and [Value Help](#) level.

- **Write**

- The default value access category when a role is created is [No Access](#). This means that this business role has no write authorizations at all (display only). You can add specific authorizations ([Restricted](#)) or in cases where you want to grant full access for all restriction types and restriction fields, you can choose [Unrestricted](#) ('\*').

Switching the write access to [Restricted](#) allows you to define which data can be edited by the users assigned to this business role.

- In the [Values](#) area, you can define the authorization values for the desired restriction fields.

If you don't want to grant access to a restriction field on purpose, you can choose the status [Not maintained](#).

- Every authorization you define in the [Write](#) access category is inherited to the [Read](#) and [Value Help](#) access category.

- **Read**

- The default status of the access category [Read](#) is [Unrestricted](#).

Switching the read access to [Restricted](#) allows you to define which data can be seen by the users assigned to this business role.

In the [Values](#) section, you can define the instance-based restrictions for the desired restriction fields used for value helps.

For more information about the [Values](#) area, please refer to the [Write](#) section above.

- Every authorization you define in the [Read](#) access category is inherited to the [Value Help](#) access category.

- **Value Help**

- You can define authorizations for value helps that are used in a business role. These value help authorizations will not influence the defined restrictions for read access.

In the context of a business role, you can authorize the value help access, for example to business partners that belong to certain authorization groups.

- [Leading Restriction](#)
- In this section, we will describe how you can reduce the maintenance effort of restriction fields that are used in multiple restriction types of an access category.
- Restriction types that contain general organizational restriction fields can be found in the top section called [General](#). These restriction types contain only one restriction field.

The status of the restriction fields that belong to this section can be changed to *Leading Restriction*. You can do this by selecting the *Leading Restriction* checkbox. This status is visible through the *Leading Restriction* symbol in the *Restrictions Overview*. The respective symbol looks like this:



That means the value in this field is automatically inherited to other restriction types the field is used in as well.

You want to, for example, define that the values for the country templates for Austria and Switzerland are applied in all restriction types the *Company Code* field is used in. So you select the values **AU01** (for Austria) and **CH01** (for Switzerland) and switch *Leading Restriction* on. Then these values are automatically inherited to all occurrences of the *Company Code* field.

- **Default Values from Business Catalogs**

Business catalogs can contain default values. In rare cases, these might overwrite the values defined by you in the Restrictions and Values section. The system then displays a message listing the default values that are added from the business catalog. To avoid this, you then need to remove the respective business catalog from the business role. The respective symbol that is visible in the *Restrictions Overview* looks like this:



- 

## Related Information

[Display Restriction Types \[page 2685\]](#)

[Maintain Restrictions in Business Role](#)

[Maintain Restrictions UI \[page 2674\]](#)

[Restrictions Overview \[page 2676\]](#)

## 6.3.1.14.9.5.1 Maintain Restrictions UI

### Maintain Restrictions

The *Maintain Restrictions UI* combines all restrictions of a restriction type of all access categories per business role in one place. It includes a main view showing all access categories and assigned restriction types and a detail view where you can change the assigned restrictions and values if required.

#### Main View

At the top of the main view, name and ID of the business role and the defined settings of the access categories are displayed. You can change these if required in the dropdown lists shown directly below. Underneath, the assigned restriction types are listed.

The *More Actions* area in the header includes the following options:

- *Maintain Empty Restrictions*

When you are in the edit mode you can apply the selected field settings to all empty restrictions of a selected access category.

- *Display Restrictions Overview*

You can navigate to the *Restrictions Overview* from here and from the object view in both edit and display mode

The following colour scheme of the assigned restriction types helps you to organize your work:

- Green: all restrictions of this type have been processed (either maintained with values or set to *Not Maintained* intentionally)
- Orange: at least one field is still empty  
Please note that there is one exception to this colour scheme. If a restriction is not directly maintained for a specific access category but filled via inheritance from a higher access category it is shown as green even if it was not processed manually.

The restriction type you're currently working on is displayed under focus.

### Detail Area

The detail area consists of the following three tabs:

- Values
- Description (if available)
- Business Catalogs

You can carry out the following activities in the detail view:

- Display all restrictions and values of the restriction type that is currently under focus.
- Edit restrictions and values by clicking the pencil icon when you are in edit mode.
  - Maintain the field settings *Restricted*, *Unrestricted Access* or *Not maintained* by selecting the radio buttons. The option *Maintained in Derived Role* is only available if the business role you're currently editing is a leading business role.
  - Select and deselect values from the value help list
  - Add number ranges if number ranges are supported  
Edit or paste multiple values in a comma-separated form in the text area

#### ⓘ Note

Please note that any change in the text area will be processed once you leave the text area.

- Display an overview of all assigned values

#### ⓘ Note

Mass maintenance is supported as well. You do this by selecting multiple restrictions and then clicking on one of the following buttons in the header of the restrictions list:

*Unrestricted Access*

*Not maintained*

*Maintained in derived role* (this option is only available for leading business roles)

- Easily change the access category of a restriction by selecting the following radio buttons: *Write*, *Read*, *Value Help*, *Read*, *Value Help*, or *Value Help*.
- Add other instances of this restriction type of the highest access category (which is automatically selected) by pressing the the *Add* button in the header of the restrictions list.
- Define a restriction of the *General type* as leading by selecting the *Leading Restriction* checkbox. If required, you can define several general restrictions as leading. You can do this by selecting the required restrictions and then clicking *Leading*.
- Remove restrictions if required by selecting the restrictions and clicking the the *Remove button* in the header of the restrictions list.
- Hide fields you have set as *Unrestricted*. This makes it easier for you to gain an overview of all restrictions. You can do this clicking *Hide Unrestricted* in the very top right corner of the screen.

### 6.3.1.14.9.5.2 Restrictions Overview

#### Restrictions Overview

You can use the *Restrictions Overview* to display all restrictions of different access categories and values that are maintained for the business role. The following filtering options are available:

- *Restriction Field*
- *Restriction Type*
- *Restriction Type ID*
- *Access Category*
- *Field Settings*

Clicking on a restriction leads you directly to the maintenance screen where you can make adjustments if required. You can also open the maintenance screen by clicking *Maintain Restrictions* in the top right corner of the screen.

## 6.3.1.14.9.6 How to Download and Upload Business Roles

Download business roles from the source system and then upload them to the target system to make them available there.

### Context

You can download one or more business roles from the source system and then upload them to the target system using an `XML` file.

### Procedure

1. To download the required business roles, go to your source system and select them.
2. Click *Download* and save the `XML` file on your hard drive.
3. To upload the required business roles, go to your target system and click *Upload*.
4. Browse for the `XML` file and click *OK*.

#### ⚠ Caution

Please do not modify the XML file on your hard drive before uploading it to the target system. This can lead to technical issues.

#### ⓘ Note

Some business catalogs with tenant-specific functions might not be in scope in all target tenants. For example, the business catalog *Extensibility - Transport Management - Export (SAP\_CORE\_BC\_SL\_EXP)* provides authorizations for exporting key user extensibility. As you can't export key user extensibility transports from the production system (but only import them), the business catalog may not be available in your production system.

## 6.3.1.14.9.7 How to Create Leading and Derived Business Roles

### Context

If you need to create several business roles with common characteristics, you can define one business leading role and then derive other business roles from it. This leading role contains the basic settings such as access

restrictions, the assigned business catalogs and common restrictions, such as *General Accountant* or *General Planner*. The values defined in the leading business role can't be changed in the derived business role. You can, however, define additional values for the derived role.

## Procedure

1. Open the business role you want to define as master business role and click *Edit*.
2. Select the *Is Leading Business Role* checkbox and click *Save*. The business role now has a *LEADING BUSINESS ROLE* label.
3. Go back to the list view, select this master business role and click *Create Derived Business Role*.
4. The system suggests a business role ID you can enhance.
5. Enter a role description and click *OK*. The system displays the business role maintenance screen. The business role has a *DERIVED BUSINESS ROLE* label.
6. Add further values to the derived business role as required.
7. Save your entries.

## 6.3.1.14.9.8 How to Compare Business Roles

### Context

To display the common features and the differences between business roles, you can compare them in the *Maintain Business Roles* app. On the *Compare Business Roles* screen, you can display the restrictions, business users, business catalogs and launchpad spaces, which the business roles have in common, or those that the business roles don't have in common. You can only compare a maximum of two business roles. They can be active or in draft mode.

## Procedure

1. On the *Maintain Business Roles* initial screen, select the two business roles you want to compare.
2. Click *Compare*. The system displays the *Compare Business Roles* screen for these two business roles including information about the assigned restrictions, business users, business catalogs and launchpad spaces.

If the *Write*, *Read* or *Value Help* access category is restricted in both roles, you can navigate further to display the assigned restriction types for that access category.

### Restrictions

You have the following options for comparing restrictions:

***Display All***

Display all restriction types assigned to one of the roles or to both.

***Display Same Restriction Types with Same Values***

Display the restriction types that are assigned to both roles and have the same values in both roles.

***Display Same Restriction Types with Different Values***

Display the restriction types that are assigned to both roles and have different values in each role.

***Display only Differences***

Display only the differences between the two roles, for example if a restriction type is only assigned to one role but not to the other role.

The restriction types are coloured as follows:

Colour	Explanation
Green	The restriction type is assigned to both roles and in both roles, the values are the same.
Orange	The restriction type is assigned to both roles but the values are different.
Red	The restriction type is only assigned to one role but not to the other.

If the restriction types are colored in green or orange, you can navigate further and display more details.

The system displays the restriction types assigned to the roles you want to compare next to each other. If there are differences between the two roles, the order of the instances reflects the degree of similarity. You can use drag and drop if you want to compare two other restriction types than the ones suggested by the system.

***Business Users, Business Catalogs and Launchpad Spaces***

You have the following options for comparing business users, business catalogs and launchpad spaces:

***Display All***

Display all business users, business catalogs and launchpad spaces that are assigned to one of the roles or both.

***Display Only Common Business Users / Display Only Common Business Catalogs / Display Only Common Launchpad Spaces***

Display only the business users, business catalogs or launchpad spaces the two business roles have in common or that are assigned to both.

***Display only Different Business Users / Display only Different Business Catalogs / Display Only Different Launchpad Spaces***

Display only the business users, business catalogs or launchpad spaces that differ from one business role to the other or that are assigned to one business role but not to the other.

## 6.3.1.14.9.9 How to Create Spaces and Pages for a Business Role

### Context

When creating a new business role, you can create a new launchpad space, use an existing launchpad space or launchpad spaces based on a predefined space. If required, you can add multiple spaces to the role by clicking *Add* on the *Assigned Launchpad Spaces* tab. If a launchpad space is no longer relevant for a role, you can click *Remove*. Please note that you can assign an existing launchpad space to several business roles at once. For more information, see the *Related Information* section.

#### Note

The group-based home page is deprecated. It will be removed in a future version. For new systems (starting with SAP S/4HANA Cloud 2302), the group-based home page is no longer available. It will be replaced with the spaces and pages mode. Therefore, a warning message is shown. It appears when you save a business role without an assigned space. It reminds you that you need to assign a space to make the tiles visible for the users when you have already switched to the spaces and pages mode.

For more information, see SAP Note [2970113](#).

### Related Information

[Step by Step: Create a New Space and Page for a Business Role](#)

[How to Make Mass Changes to Business Roles \[page 2683\]](#)

## 6.3.1.14.9.10 How to Transport Business Roles

### Context

You can export business roles from your quality system to your productive system using customizing transports.

#### Note

- Once you have transported a business role, no change documents will be written for this business role in the productive system. Change documents for transported business roles are only available in the quality system.

- If you transport a derived business role, the leading business role and all other derived business roles need to be added as dependencies to the transport request as well.  
If you transport a leading business role, all derived business roles need to be added as dependencies to the transport request as well.
- Please note that business roles that are pre-delivered by SAP (business roles whose ID starts with SAP\*) cannot be transported.

## Procedure

1. In the *Business Roles* overview screen of the *Maintain Business Roles* app, select one or more business roles.
2. Click *Add to Transport*. A list of available customizing transports appears.

### ⓘ Note

The default transport is always displayed as well as your own transports. If no transport exists, a confirmation dialog is shown. If you confirm, a new transport is created.

3. Select the required transport and click *OK*. The selected business roles are then added to the transport.
4. Open the *Export Customizing Transports* app and release your request. For more information, see [Export Customizing Transports \[page 2582\]](#).
5. Import your business role to the target system:
  - a. Open the *Manage Software Components* app.
  - b. Select *Business Configuration* in the *Type* dropdown and click *Go*.
  - c. Select the component to which the relevant configuration was exported.
  - d. In the detail view of the component, check out the change that contains your business role(s). For more information, see [How to Work with Branches](#).

### ⓘ Note

If you want to delete a business role that was previously transported or is currently included in an open transport request, the system offers you to add it directly to an available transport.

If the business role was previously transported and there is currently no transport available, a confirmation dialog is shown. If you confirm, a new transport is created.

## Related Information

[Export Customizing Transports \[page 2582\]](#)

[Working in the Export Customizing Transports App \[page 2584\]](#)

[Manage Software Components](#)

[How to Delete Previously Transported Business Roles \[page 2682\]](#)

## 6.3.1.14.9.10.1 How to Delete Previously Transported Business Roles

### Context

If you want to delete a business role that was previously transported or is currently included in an open transport request, the system offers you to write it directly to an available transport.

### Procedure

1. In the *Business Roles* overview screen of the *Maintain Business Roles* app, select one or more business roles you want to delete.
2. Click *Delete* and confirm the dialog. A list of available customizing transports appears.

#### ⓘ Note

The default transport is always displayed as well as your own transports. If no transport exists, a confirmation dialog is shown. If you confirm, a new transport is created.

3. Select the required transport and click *OK*. The deletion of the selected business roles is then added to the transport and the selected roles will be deleted.

#### ⓘ Note

Your selection may contain business roles that were not previously deleted. These will be deleted directly without a transport. The system also offers you to delete the business role without transporting the deletion. The role will not be deleted in the target system in this case.

4. Open the *Export Customizing Transports* app and release your request. For more information, see [Export Customizing Transports \[page 2582\]](#).
5. Import your business role to the target system:
  - a. Open the *Manage Software Components* app.
  - b. Select *Business Configuration* in the *Type* dropdown and click *Go*.
  - c. Select the component to which the relevant configuration was exported.
  - d. In the detail view of the component, check out the change that contains your business role(s). For more information, see [How to Work with Branches](#).

### Related Information

[Export Customizing Transports \[page 2582\]](#)

[Working in the Export Customizing Transports App \[page 2584\]](#)

## 6.3.1.14.9.11 How to Make Mass Changes to Business Roles

### Context

You can use the mass change wizard to change multiple business roles at once. This is helpful, for example, if you have switched from using custom spaces to predefined spaces or if you start using spaces. In that case, you can select all affected business roles and choose *Assign Predefined Spaces*. The system then automatically assigns predefined spaces to all of the business roles in question. If you choose *Remove Launchpad Spaces Already Assigned*, the custom spaces that were previously assigned to the business roles are automatically removed.

#### ⓘ Note

Please note that SAP-defined business roles can only be modified for business user assignment in the initial setup. Apart from this, they can be updated by SAP only. We recommend that you create your own business roles from templates so that you can modify them as required.

Select the business roles you want to change and click *Mass Change*. The *Mass Change Wizard* appears. The pattern of the wizard is described in the next section.

#### ⚠ Caution

If you have selected a combination of leading, non-leading, and derived business roles, some attributes might not be selectable. For example, business catalogs in derived business roles can't be edited. Therefore, business catalogs are not included in the mass change if a derived business role is selected.

To change all of the listed attributes, select only leading business roles.

### Procedure

1. Select the required attributes. You can make the following changes:

Area	Attributes
Business Role Data	Role Group
	Expose to SAP BTP
	Inherit Spaces in Derived Business Roles

Area	Attributes
<b>Business User Assignment</b>	Add Business Users Remove Business Users
<b>Business Catalog Assignment</b>	Add Business Catalogs Remove Business Catalogs
<b>Launchpad Space Assignment</b>	Add Launchpad Spaces Remove Launchpad Spaces
<b>Assign Predefined Spaces</b>	Assign predefined spaces your business roles  You see the results and can filter for error or success statuses in the <a href="#">Mass Change Overview</a> .
<div style="border: 1px solid #ccc; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>Please note that this option is only relevant for business roles that were created from business role templates.</p> </div>	
<b>Access Categories</b>	Change access categories as required.
<b>Restrictions</b>	Select the required access category, such as <a href="#">Value Help</a> and select the required restriction change, such as <a href="#">Add Restrictions</a> .
<b>Copy Restrictions</b>	<ol style="list-style-type: none"> <li>1. Click <a href="#">Add Restrictions</a> or <a href="#">Remove Restrictions</a>.</li> <li>2. Copy the required restrictions using the <a href="#">Copy Restriction Values From</a> and the <a href="#">Copy Restriction Values To</a> value help.</li> <li>3. Select additional access categories if needed.</li> </ol> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <p><b> ⓘ Note</b></p> <p>You can use this option in case a restriction type is deprecated to copy restrictions from the deprecated restriction type to the successor restriction type(s). You can either add the copied restriction values in the fields to existing ones or remove all and replace them with the copied restrictions.</p> </div>

2. Apply the required attributes to your business roles.
3. Review and confirm your changes.

## Related Information

[Assigning Spaces to Several Business Roles at Once](#)

## 6.3.1.14.10 Display Restriction Types

You can use this app to display restriction types and their validity.

With this app you can display restriction types, the assigned fields, and in which business catalog the restriction type is used.

### Key Features

You can use this app to:

- Display all available restriction types and their fields.
- Check whether the restriction fields support number ranges.
- Display how these restriction types can be used in which business catalogs.
- Display component: check under which component you can create an incident if required.

Restriction types bundle one or more restriction fields. The restriction type *Organizational Area*, for example, contains the following fields: *Purchasing Group*, *Purchasing Organization*, *Division*, *Sales Organization*, *Distribution Channel*.

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## 6.3.1.14.11 Manage Business Role Changes After Upgrade

With this app you can display all relevant changes to business catalogs and restriction types after an upgrade. If a new restriction type was added to a business catalog for example, you can maintain the corresponding restrictions using this app.

## Key Features

You can use this app to:

- Display a list of changed restriction types
- Display details of the change
- Filter for certain change types (such as *Added* or *Removed*)
- Display the affected business catalogs and their dependencies
- Display deprecated business catalogs, their successors and adopt the respective changes
- Display the number of affected business roles
- Cross-navigate to the affected business roles in the *Maintain Business Roles* app to view further details
- Mass-maintain restrictions of the same access categories

## Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## Related Information

[How to Manage Changed Restriction Types After an Upgrade \[page 2688\]](#)

[How to Manage Changed Business Catalog Dependencies After an Upgrade \[page 2689\]](#)

[How to Manage Deprecated Business Catalogs After an Upgrade \[page 2690\]](#)

## 6.3.1.14.11.1 Manage Business Role Changes After Upgrade: Background

It is important for you to know how the quarterly release changes impact your productive business roles.

Business role changes often have the technical reason that new developments may require changes to existing applications, business catalogs, restriction types and/or restriction fields. The affected business roles need to be adapted accordingly to ensure that existing apps behave consistently before and after the upgrade.

The following typical changes might occur:

- Business catalogs were deprecated or split, or have new dependencies so new business catalogs need to be assigned.
- New apps were added to existing business catalogs.
- New restriction types were added, removed, or access categories were changed.

## 6.3.1.14.11.2 Best Practices for Managing Business Role Changes After an Upgrade

We recommend downloading a list of all the changes that have taken place after an upgrade before you adjust your business roles using the [Manage Business Roles After an Upgrade](#) app. You can do this by clicking [Download](#) (csv format is supported). This helps you to keep an overview of what has changed. Once you have modified your business roles, the respective changed objects disappear from the list in the app. It is therefore advisable to keep a separate worklist on your local machine.

We also recommend adjusting your business roles using the [Manage Business Roles After an Upgrade](#) app only if one change applies to one business role. If there is more than one change, we recommend making the required changes manually in the [Maintain Business Roles](#) app.

### Restriction Types / Business Catalog Dependencies

An affected business role is no longer shown in the list once the business role has been saved the first time after the upgrade/update.

The change itself will no longer be visible in the app after all listed affected business roles have been saved the first time after the upgrade/update.

### Deprecated Business Catalogs / Business Role Template

An affected business role is no longer shown in the list once the respective change has been incorporated in the business role.

The change itself will no longer be visible in the app after the respective change has been incorporated in all affected business roles.

### 6.3.1.14.11.3 How to Manage Changed Restriction Types After an Upgrade

#### Context

If you want to have transparency over all of the changes to restriction types after an upgrade, and you want to maintain the corresponding restrictions in your business roles, proceed as follows:

#### Procedure

1. Select the [Manage Business Role Changes After Upgrade](#) tile on the SAP Fiori Launchpad to open the app. On the initial screen, view the [Restriction Types](#) area to get an overview of what has changed with the last upgrade. Filter for certain change types if required, for example if you only want to see the removed or the added restriction types.
2. Download a csv file containing all recently changed restriction types if required.
3. Click on the required restriction type to check how the changed restriction types affect your business roles. The business roles affected by the changes to this restriction type are displayed in a list. By clicking on the restriction type [Name](#) above the list, you can view a description of the restriction and a list of business catalogs in which this restriction type has been changed (for example added or removed). Please note that any derived roles will be filtered out from the list of the affected business roles to allow for synchronization with the master role.
4. To find more information about the affected business roles or to make further changes, you can cross-navigate to the [Maintain Business Roles](#) app by clicking on the link listed in the [Business Role ID](#) column. When you click on the [Information](#) pushbutton next to the link, a list of all other changes that have affected the business role after the upgrade and that might be relevant for you as well is displayed.
5. To make the necessary changes in your business roles, select the required business role and click [Maintain \(Write, Read, Value Help\) Restrictions](#). Restrictions can be mass-maintained for multiple business roles if the roles have the same access categories. If they have different access categories, the pushbutton is disabled. On the [Maintain Restrictions](#) screen, you can also view the related business catalogs and the business roles selected on the previous screen. You see the restriction fields of the restriction you are currently maintaining. To see other restrictions these business roles might include, click the glasses icon ([Display Restrictions](#)) to cross-navigate to the [Maintain Business Roles](#) app.
6. Click the [Edit](#) icon and assign the required values or ranges. Click [Not maintained](#) if you want to make clear that you have not maintained a field on purpose and not by accident.
7. Click [Apply](#).

#### Results

The system displays the results of the update in a pop-up. If the update has been successful, a green icon is visible. If issues occurred, a warning message is displayed.

When you close the pop-up, you will be directed to the main view again.

## 6.3.1.14.11.4 How to Manage Changed Business Catalog Dependencies After an Upgrade

### Context

If you want to have transparency over all the changes to business catalog dependencies after an upgrade, proceed as follows:

### Procedure

1. Select the *Manage Business Role Changes After Upgrade* tile on the SAP Fiori Launchpad to open the app. On the initial screen, view the *Business Catalog Dependencies* area to get an overview of what has changed with the last upgrade. Filter for certain change types if required, for example if you only want to see the removed or the added dependencies.
2. Download a csv file containing all recently changed business catalog dependencies if required.
3. Click on the required business catalog dependency to check how the changed dependency affects your business roles. The business roles affected by the changes to this dependency are displayed in a list. By clicking on the business catalog *Name* above the list, you can view a description of the business catalog and its newly added dependencies or recently removed dependencies after an upgrade.
4. To find more information about the affected business roles or to make further changes, you can cross-navigate to the *Maintain Business Roles* app by clicking on the link listed in the *Business Role ID* column.

## 6.3.1.14.11.5 How to Manage Deprecated Business Catalogs After an Upgrade

### Context

If you want to view all deprecated business catalogs after an upgrade, you want to assign the successor business catalogs and adopt the relevant changes in your business roles, proceed as follows:

### Procedure

1. Select the *Manage Business Role Changes After Upgrade* tile on the SAP Fiori Launchpad to open the app. On the initial screen, view the *Deprecated Business Catalogs* area to get an overview of what has changed with the last upgrade.
2. Download a csv file containing all recently deprecated business catalogs if required.
3. Click on the required business catalog to check how the deprecation affects your business roles. The business roles affected by the deprecated business catalog are displayed in a list. By clicking on the business catalog *Name* above the list, you can view a description of the deprecated business catalog and the successor business catalog.
4. Select the required business role and click *Adopt Changes* if you want to take over the changes to your business roles. Please note that you might lose your maintained restrictions by doing this. We recommend that you validate the business role after the automatic adoption. By adopting the changes, you add all the successors business catalogs to the selected roles. The deprecated catalog will be removed.

#### ⓘ Note

If the deprecated business catalogs contain extensions and you click *Adopt Changes*, the included extensions will be removed. If you want to keep these extensions, you need to re-assign them.

5. Select the *Add the dependent business catalogs of the successor business catalogs* checkbox if required.
6. Click *OK*. The system displays a progress bar and a list of change results when it is done.
7. Click *Close* to return to the *Deprecated Business Catalogs* area.

## 6.3.1.14.11.6 How to Manage Business Role Template Changes After an Upgrade

### Context

To get an overview of changes in the business role template area since the last upgrade, you can display any differences between business role and business role template in the [How to Manage Business Role Changes After an Upgrade](#) app. Either the template or the role that was based on the template might have changed. For example, business catalogs might have been added or removed. To apply these changes in a detailed maintenance screen, you can cross-navigate to the [Business Role Templates](#) app.

### Procedure

1. Select the [Manage Business Role Changes After Upgrade](#) tile on the SAP Fiori Launchpad to open the app. On the initial screen, the [Business Role Templates](#) area provides an overview of what has changed with the latest upgrade.
2. Download a csv file containing all recently changed business templates if required.
3. Click on the required business role template for details of the changes. The type of change (for example [Different from Business Roles](#)) is already displayed on the initial screen.
4. To display a detailed comparison, select one or more affected business roles and click [Compare with Business Role Template](#). This takes you to the [Business Role Templates](#) app.
5. All deviations of the business roles from the business role templates are displayed in a table. You can click [Apply All](#) if you want to take over all of the changes. Alternatively, you can go through the list and select deviations you would like to apply.

You can define one role as leading and all changes you make to this role will be automatically transferred to the roles you selected in the [Deviations](#) table.

6. Click [Save](#) when you're done.

Please note that you can also adjust your businesss roles directly in the [Business Role Templates](#) app.

### Related Information

[Business Role Templates \[page 2702\]](#)

## 6.3.1.14.11.7 Phase-In / Phase-Out Status

If a restriction is in status *Phase-In*, the values have no effect in the system at the moment. Therefore, you can prepare your system for new restrictions that will be active in the next release.

The *Phase-Out* status denotes that the restriction will be removed in the next release.

## 6.3.1.14.12 Display Technical Users

This app shows all technical users that exist in the system. To call the app, log on to your SAP Fiori Launchpad and go to [Identity and Access Management](#) [Display Technical Users](#)

### Purpose

This app has two tabs:

- Technical User
- SAP Support User Request Log

In the *Technical User* tab, you can display technical users that can be services that are used to automate technical tasks in the system, for example, a print queue user who pulls print jobs remotely. In addition, the service and support users of the software provider or hosting provider are technical users.

In the *SAP Support User Request Log* tab, you can display more information about when and why SAP support users accessed your customer system in the past 12 months. For each support user, the relevant incident ID, access level, access category, customer user, request date and validity date is displayed when you click on the required entry in the *Users* list. SAP support user IDs are pseudonymized to respect the data subject rights of SAP employees according to GDPR.

For more information about access levels and access categories, see the *Related Information* section.

### Key Features

You can use this app to:

- Lock or unlock the following types of technical users: Print users and communication users
- Change the user name and password of some types of technical users
- Display details of which support users were created for the system
- Display all technical users in the system
- Export user data (such as *User ID* or *User Group*) to a spreadsheet if required

## Note

To lock and unlock communication users, use the [Maintain Communication Users](#) app. See section Related Information.

All users are assigned to one of the following user groups:

### User Groups

Group	Explanation
<b>SAP Technical Users</b>	These users are predefined by SAP to run tasks in the system for cloud operation activities and automated processes. SAP technical users are also required for customer-specific tasks, such as running and scheduling jobs. The required job template contains the SAP technical user needed to run the job by default.
<b>Communication Users</b>	In this group, all communication users used in communication arrangements that are managed by customers are displayed.  Communication users are technical users required for communication between your system and any integrated system. The following types of communication users are available: <ul style="list-style-type: none"><li>• SAP communication users: Predefined by SAP</li><li>• Communication users: Created by customers Custom communication users are created in the <a href="#">Maintain Communication Users</a> app and are assigned to systems for inbound and outbound communication. These technical users either use basic authentication or certificates to communicate between your system and the integrated system. The communication system, together with a communication scenario, is assigned to a communication arrangement. Thus, all communication scenario roles are assigned to the communication user which provides this user with all permission required for the connection between your system and the integrated system.</li></ul>
<b>Support Users</b>	SAP uses support users to provide customers with support if issues occur in the system. These users only exist temporarily and are deleted by SAP after the support process is completed.

## Supported Device Types

- Desktop
- Tablet

## SAP Technical Users

SAP*	This user has always been used in on-premise systems to log on to newly created tenants. This user is locked in your tenant and it's not used for any system operation.
SAP_CUST_BUS	This user is a reference user assigned to all customer business users and SAP support users. General authorizations required for basic system access are assigned through this reference (for example to access Fiori Launchpad). The user can't be used by itself. It's locked and has no logon credentials.
SAP_LMADM	This user is used for system internal, automated software lifecycle management across all tenants. It's used in processes that need to update tenant-specific data after changes to tenant-independent data. This user is unlocked. External system logon isn't possible.
SAP_SMTP_IN	This user is used as the SMTP mail server user.
SAP_SPC	This user is used for SAP-internal automated system operation.
SAP_SYSTEM	This user is the default user to run system-internal technical jobs.
SAP_WFRT	This user is used by Business Workflow for the automated background processing of work items.
SAP_WSRT	This user is used for system internal processing in the Web service runtime.

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.12.1 SAP Support User Request Log

In the *Display Technical Users* app, you can display more information about when and why SAP support users accessed your customer system in the past 12 months. For each support user, the relevant incident ID, access level, access category, customer user, request date and validity date is displayed when you click on the required entry in the *Users* list. SAP support user IDs are pseudomized to respect the data subject rights of SAP employees according to GDPR.

## Related Information

[Access Levels \[page 2695\]](#)

[Access Categories \[page 2697\]](#)

### 6.3.1.14.12.1.1 Access Levels

Authorizations of SAP support users

Access levels define the authorizations of SAP support users when accessing customer systems.

Access Levels

Name	Use
<a href="#">SUPPORT_DEFAULT</a> (Platform Analysis, Display Configuration, No Business Authorization)	The <a href="#">SUPPORT_DEFAULT</a> access level provides display access to ABAP Platform transactions, display of Customizing tables and views, display of system tables with content delivered by SAP or transported in customer systems. It also provides authorization to display, activate and deactivate transient logs and traces without access to payload data. It grants authorization for the debugging of the own user and external debugging of other users. Authorization to change values of variables or to alter the code flow in the debugger is not granted. In case business authorizations are required for support, they are agreed in customer support cases and are copied from a specific customer business user or customer communication user as additional authorization.
<a href="#">SUPPORT_DEFAULT_APP</a> (Platform Analysis, Display Configuration, Display Business Data)	The <a href="#">SUPPORT_DEFAULT_APP</a> access level inherits all the authorizations from the <a href="#">SUPPORT_DEFAULT</a> access level. Additionally, it grants display access to application tables and selected business transactions that enable analysis. It includes display of most business application logs as well as display, activation and deactivation of transient traces with access to payload data. Authorization to change customer data is not granted.
<a href="#">SUPPORT_EXTENDED</a> (Platform Analysis, Limited Administration, Full Business Authorization)	The <a href="#">SUPPORT_EXTENDED</a> access level inherits all the authorizations from the <a href="#">SUPPORT_DEFAULT_APP</a> access level. Additionally, it grants full business application authorization. It includes all backend transactions and reports of the application layer. It grants limited ABAP Platform administration authorization, such as deletion of caches. Authorization to change system configuration, Customizing data, or development objects is not granted.

Name	Use
<a href="#">SUPPORT_CUSTOMIZING</a> (Limited Platform Customizing, Full Business Customizing)	The <a href="#">SUPPORT_CUSTOMIZING</a> access level inherits all the authorizations from the <a href="#">SUPPORT_DEFAULT_APP</a> access level. Additionally, it grants authorization to maintain selected ABAP Platform Customizing as well as unrestricted business application Customizing.
<a href="#">SUPPORT_CONTENT_ACT</a> (Content Activation)	The <a href="#">SUPPORT_CONTENT_ACT</a> access level inherits all the authorizations from the <a href="#">SUPPORT_EXTENDED</a> and the <a href="#">SUPPORT_CUSTOMIZING</a> access levels. Additionally, it grants administrative authorization to the content framework which manages the Customizing content lifecycle.
<a href="#">SUPPORT DEVELOP LOCAL</a> (Local Development)	The <a href="#">SUPPORT DEVELOP LOCAL</a> access level inherits all the authorizations from the <a href="#">SUPPORT_EXTENDED</a> access level. Additionally, it grants authorization to create and execute local development objects. This access level may need to be used to support frameworks that generate local development objects in the customer name space (such as extensibility or data migration). It allows to test execute all function modules and static methods. It also allows changing field values and altering the code flow in the debugger. Authorizations to modify development objects delivered by SAP are not granted.
<a href="#">SUPPORT DEVELOP</a> (Unrestricted Development)	The <a href="#">SUPPORT DEVELOP</a> access level inherits all the authorizations from the <a href="#">SUPPORT DEVELOP LOCAL</a> access level. Additionally, it grants authorization to modify development objects delivered by SAP. Standard procedure for such changes is a hotfix or an emergency patch. This access level may only be used in emergency situations where that process is not applicable.
<a href="#">SUPPORT_USER_ADMIN</a> (User and Role Administration)	The <a href="#">SUPPORT_USER_ADMIN</a> access level inherits all the authorizations from the <a href="#">SUPPORT_DEFAULT</a> access level. Additionally, it grants authorization for local user and role administration. This access level may be used for support cases related to business user or communication user management.
<a href="#">SUPPORT_SYSTEM_CONFIG</a> (System Configuration)	The <a href="#">SUPPORT_SYSTEM_CONFIG</a> access level inherits all the authorizations from the <a href="#">SUPPORT_EXTENDED</a> access level. Additionally, it grants authorization to manually configure SAP managed communication scenarios. Automatic setup is mandatory for such communication scenarios in customer systems. This access level is used in situations where automatic setup failed.

Name	Use
<a href="#">SUPPORT_SYSTEM_ADMIN</a> (Unrestricted System Administration)	The <a href="#">SUPPORT_SYSTEM_ADMIN</a> access level grants unrestricted administrative access. It includes authorization profile SAP_ALL. Cloud operations may need to use it for system lifecycle management. Support teams may need to request it as the final escalation of privileges for support cases that cannot be resolved with other access levels. Root cause analysis is mandated for such support cases. The root cause must be resolved.

## Related Information

[SAP Support User Request Log \[page 2694\]](#)

[Access Categories \[page 2697\]](#)

### 6.3.1.14.12.1.2 Access Categories

Access categories classify the purpose of SAP support user access to customer systems.

Access Categories

Name	Use
<a href="#">Operations</a>	The <a href="#">Operations</a> access category is used for system access by central cloud operations teams for system lifecycle management, reactive system management and proactive system management. Such access is not based on individual support cases. Taking over authorization from customer business users or customer communication users is not possible.

Name	Use
<i>Customer Support</i>	The <i>Customer Support</i> access category is used for system access based on support cases raised by the customer. The purpose of the system access is documented in a customer support case and can be reviewed in customer service management. System access is restricted to a processor of a customer case that is not completed. Support cases created for productive systems also allow access to non-productive systems. Support cases created for non-productive systems do not allow system access to productive systems. System access to a productive system requires a case for that specific system. Customers may allow the processor to copy authorizations from one specific customer business user or customer communication user. Consent to that is documented in the customer support case. The processor of the case can then copy these authorizations in addition to a support access level. The <i>SUPPORT_DEFAULT</i> and <i>SUPPORT_DEFAULT_APP</i> support access levels are automatically approved for processors of customer support cases to access customer systems. The <i>SUPPORT_EXTENDED</i> support access level is also automatically approved for processors of customer support cases to access non-productive customer systems. To access productive customer systems using the <i>SUPPORT_EXTENDED</i> support access level, an SAP internal approval process is in place for SAP S/4HANA Cloud as well as SAP BTP ABAP environment. Any requests for higher privileged support access levels in customer systems enforce the principle of dual control using SAP internal approval processes.
<i>Health Check Support</i>	The <i>Health Check Support</i> access category is used for system access based on support incidents raised internally by automatic health checks. The purpose of the system access is documented by SAP internally via the incidents. Procedures for approval of access levels match the procedures of the <i>Customer Support</i> access category. The same is true for restrictions of access to productive systems versus non-productive systems.
<i>Emergency Support</i>	Support user access is controlled by a platform for central cloud management. The <i>Emergency Support</i> access category is used for locally created support users in situations where the connection to the central cloud management is interrupted. The purpose of the emergency support access is documented in a customer support case and can be reviewed in customer service management.

Name	Use
<a href="#">SAP Internal Support</a>	The <a href="#">SAP Internal Support</a> access category is used for system access to SAP internal test or demo systems. It is not intended for system access to customer systems. We recommend that you create an incident for cloud operations to validate the configuration of support user access if there is any such access in customer systems.

## Related Information

[SAP Support User Request Log \[page 2694\]](#)

[Access Levels \[page 2695\]](#)

### 6.3.1.14.13 Business Catalogs

You use this app to display all available business catalogs.

With this app you can get an overview of the business catalogs, their status (for example [Deprecated](#)), and their usage within business roles. You use this application to see which applications and business catalogs delivered by SAP have changed, for example after an upgrade. They may also be deprecated. As a key user, you need to have an overview of these changes to adapt the affected business roles accordingly.

## Key Features

You can use this app to:

- Display business catalogs, their usage in roles, and general description
- Display business catalog changes: Check if the business catalogs or apps in your area have changed (for example, a transaction has been replaced with an SAP Fiori app), and which business roles are affected by these changes.
- Display assigned apps: You can display apps that have their own tile access (classified as [Standard Application](#)) as well as ones that can be started indirectly via app-to-app navigation (classified as [Standard Navigation Target](#)).
- Display deprecations: Check if any of the business catalogs in your area are deprecated and which successors they will be replaced with. Select the affected business role and transfer the changes.
- Display dependencies: Check if a business catalog depends on other business catalogs or if other business catalogs depend on a business catalog in order to be fully operational.
- Display scope items: check which scope items the business catalogs depend on. Business catalogs that do not depend on any scope items are always visible in the system. For these business catalogs, [Scope Items \(0\)](#) is displayed on the relevant tab and the following message appears in the table: [The business catalog is not scope-dependent](#).

- Display component: check under which component you can create an incident if required.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.13.1 How to Handle Deprecated Business Catalogs

If you want to check how many deprecated business catalogs you still have in use, or you want to change assignments from the old, deprecated business catalogs to the new, active catalogs quickly and easily, proceed as follows:

#### Context

Due to ongoing development of new features and new apps, we need to periodically revise existing business catalogs. This means that some business catalogs are deprecated and replaced by new ones, and you need to assign business roles and business users to these new catalogs.

Rather than disappearing, deprecated business catalogs are identified as being obsolete, which allows you to identify them at a glance. You can also check how many deprecated business catalogs you still have in use with the [Business Catalogs](#) app. This app lets you change assignments from the old, deprecated business catalogs to the new, active catalogs quickly and easily.

#### ⓘ Note

Some business catalogs might be redesigned in each release. Please check the assignments for your business roles and business users and make the necessary changes to the assignments as soon as possible.

#### Procedure

1. In the [Business Catalogs](#) app, check how many deprecated business catalogs you still have in use.

You can filter the list of catalogs for the deprecated ones but the deprecated business catalogs are also marked with the appendix (deprecated) in the list of all catalogs in use.

2. Change assignments from the old, deprecated business catalogs to the new, active catalogs.

Once the deprecation of a business catalog is announced via the *Business Catalogs* app, the catalog will remain in the system for six months before being deleted. During this period you can use the old or the new business catalog. Within this timeframe you can do the replacement when it suits you best. In the *Business Catalogs* app, you can see the release in which the deprecation of a business catalog was announced.

**Example:** A business catalog is deprecated with a Cloud 2202 release. The business catalog will then be deleted with the Cloud 2208 release.

### 6.3.1.14 IAM Information System

With this app you can get an overview of business users in your system and what roles and restrictions are assigned to them.

With this app you can display information about the usage of business roles, business catalogs, business users and restrictions, and how they are related.

If you want to look up more information about a business role, derived business role, business user, business catalog, business role template or restriction, you can jump directly to the respective app by clicking the entity.

You can use this app for administrative tasks as part of your daily work.

#### Key Features

You can use this app to:

- Check the usage of the following entities and how they are related: Business role, derived business role, business role template, application, business user, business catalog, restriction, launchpad space, launchpad page.
- Check, for example, which business roles are assigned to a business user, which business catalogs and restrictions are therefore assigned to the business user and to which applications a user has access. You can also download a list of business users and business catalogs.
- Check, for example, which business roles and business catalogs are required to make a certain app visible for a business user on the SAP Fiori Launchpad.

#### Note

You can also display the scope items the business catalogs depend on. Business catalogs that do not depend on any scope items are always visible in the system. For these business catalogs, **Scope Items (O)** is displayed on the relevant tab and following message appears in the table: *The business catalog is not scope-dependent*.

- Display assigned apps: You can display apps that have their own tile access (classified as *Standard Application*) as well as ones that can be started indirectly via app-to-app navigation (classified as *Standard Navigation Target*).

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## 6.3.1.14.15 Business Role Templates

You can use this app to get an overview of the business role templates delivered by SAP.

With this app you can get an overview of the delivered business role templates, any changes included in upgrade, and whether you need to adapt your business roles to these changes. For example, after an upgrade, you can check if the business role templates have changed and are therefore different from the existing business role - a new business catalog might have been added or an existing catalog replaced by another. You can see which business roles were affected by the changes and adapt them if required.

## Key Features

You can use this app to:

- Display business role templates
- Display usage of business role templates in business roles
- Display differences between business role templates and business roles
- Check which business roles are affected by these changes
- Adapt business role to changes
- Create a business role
- Display scope items: check which scope items the business role templates depend on. Business role templates that do not depend on any scope items are always visible in the system. For these business role templates, *Scope Items (0)* is displayed on the relevant tab and the following message appears in the table: *The business role template is not scope-dependent*.
- Display component: check under which component you can create an incident if required.

## Supported Device Types

- Desktop

- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## Related Information

[How to Create a New Business Role \[page 2667\]](#)

[How to Create a Business Role from a Template \[page 2668\]](#)

## 6.3.1.14.16 IAM Key Figures

You can use this app to display the following information:

- Number of business users assigned to business roles
- Month of the business user's last log-on
- Number of locked and unlocked business users
- Validity of business users
- Number of business roles with unmaintained restrictions
- Number of business roles with unrestricted access

### ⓘ Note

You can filter for the productive business roles in your area, for example by adding a certain namespace followed by an asterisk (\*) in the *Filter by Business Role ID* field in the tile configuration.

- Business user price categories
- Business roles with default values from business catalogs

The default values are delivered automatically and can't be overwritten. This means either that an asterisk (\*) overwrites all defined values or that fixed values are added to the restrictions you have defined.

Business roles affected by these non-changeable values are displayed in red in the overview chart.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.14.16.1 IAM Key Figures - Background

The charts in the **IAM Key Figures** app support you to increase the security in your area and to reduce costs by displaying which business users are inactive and can be removed.

They also provide a good overview that can be useful before go-live to help, for example, you evaluate if the business user and role distribution in your area is ready to be used.

To define threshold values, you can use color codes for the bars in your charts (green: **Accepted**, yellow: **Warning**, red: **Critical**). You can, for example, define the colors based on certain time frames. If a user has not logged on to the system for 6 months and you want to use this point in time as a threshold value, you can determine that the corresponding bar in the chart is red.

You can also jump directly to the **Maintain Business Users** app or **Maintain Business Roles** app if you need more information about the business users or business roles that are shown in the overview charts.

### 6.3.1.14.17 Display Authorization Trace

With this app you can enable an authorization trace for a business user. This helps you to analyze if any authorizations are missing or are insufficient.

#### Key Features

You can use this app to:

- Activate or deactivate a trace
- Display authorization check results including already assigned authorizations and failed checks
- Display all business roles granting access to selected fields and values

A maximum of 10.000 data sets is possible, therefore we recommend to consider this when defining the selection criteria, especially the date range.

The following authorization check statuses are possible:

Status	Meaning
Successful	The authorization check was successful.
Failed	The authorization check failed.

Status	Meaning
Filtered	<p>When reading an object, an authorization check is taking place and certain data is filtered out defined by a DCL (Data Control Language).</p> <p><b>① Note</b></p> <p>When the <i>Authorization Check Status</i> is set to <i>Filtered</i>, no authorization check is carried out for any value. Instead, access filtering occurs. In this case, <i>No Value</i> is displayed for the corresponding restriction fields. Filtering means that the authorizations of the user for the specified restriction type/authorization object and for the specified field values are read and these authorization values are used as additional filter conditions during the selection of the business data. This way, the user only has access to the data for which he or she has appropriate authorizations.</p>

If an authorization check resulted in a *Failed* and *Filtered* status, you can check which business roles expose the affected restriction type. The root cause could be that the business user that has been checked is not assigned to the required business role or that the required value has not been maintained yet.

**① Note**

The trace entries are deleted automatically within a certain time frame.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

## 6.3.1.18 Business Catalogs for Identity and Access Management Apps

Get an overview of available business role catalogs and their restrictions.

You assign business catalogs to business roles that are assigned to business users. Business catalogs contain authorizations that define what a business user with a certain business role is allowed to do.

Business Catalogs for Identity and Access Management Apps

Business Role Catalog	Authorizations	Restrictions
<i>Identity and Access Management</i> SAP_CORE_BC_IAM	All authorizations for the IAM apps	Business catalog was set to obsolete due to potential SoD (segregation of duties) conflicts. Please use the other three catalogs listed below.
<i>User Management</i> SAP_CORE_BC_IAM_UM	Maintain Business Users  Display Technical Users	Not possible to assign business roles to the business users  None
	IAM Information System	None
<i>Role Management</i> SAP_CORE_BC_IAM_RM	Maintain Business Roles  Business Role Templates  Business Catalogs	Not possible to assign business roles to the business users  None  None
	IAM Information System	None
<i>Role Assignment</i> SAP_CORE_BC_IAM_RA	Maintain Business Users  Maintain Business Roles	Only possible to assign business roles to the business users. Not possible to change the user data.  Only possible to assign business roles to the business users. Not possible to change the business roles
	IAM Information System	None
<i>Identity and Access Management - Group Management</i> SAP_CORE_BC_IAM_GRP_PC	This business catalog enables users to create and maintain business user groups and business role groups.	None

### Related Information

[Identity Authentication](#)

[What Is Identity Authentication?](#)

## 6.3.1.14.19 Glossary for Identity and Access Management

Get an overview of the terminology used for identity and access management.

Terminology Overview

Term	Description
Access category	A category that defines what kind of access is granted to the users assigned to a business role, for example, <i>Read</i> , <i>Write</i> , or <i>Value Help</i> .
Business role	A business role provides users with authorizations to access apps.
Business role template	A pre-defined set of authorizations and assigned business catalogs you can use as a basis for creating new business roles.
Business user	Any person who can log on to the system, including administrators.
Restriction	An access limitation that is defined on business role level.  You can determine what kind of access is granted to specific objects, such as <i>company code</i> .
Restriction field	A field that is used to restrict the access to a specific business object, for example, <i>organizational area</i> .
Restriction type	An authorization entity that bundles the available restriction fields to a logical definition, for example, company code.
Leading restriction	A property that defines a certain restriction field as leading restriction. No matter in which restriction types the restriction field is being used, the field will have the same restriction assigned in all types.
Technical user	A technical user corresponds to a local or remote process which is typically part of the cloud management process (e.g. system provisioning, support) or intrinsic system processes (e.g. periodic clean-up of logs). There are technical users that belong to the software or service provider and there are technical users that belong to the customer.
IAM app	A unit that grants access to a set of services and authorizations. An IAM app can be grouped in business catalogs and assigned to business roles.

Term	Description
External app	An IAM app based on services, for example derived from SAP Fiori launchpad app descriptor items. It contains services and authorizations.

### 6.3.1.15 Message Monitoring

[Message Dashboard \[page 2708\]](#)

[Message Monitoring Overview \[page 2709\]](#)

[Message Monitoring for Integration Experts \[page 2710\]](#)

[Assign Recipients to Users \[page 2711\]](#)

[Message Monitoring \(Emergency Correction\) \[page 2712\]](#)

[SOAP Error Log, OData Error Log, and Event Error Log \[page 2713\]](#)

[Assigning the Business Catalog for SOAP Error Log, OData Error Log, and Event Error Log to Your Business Role \[page 2714\]](#)

[Automatic Reprocessing \[page 2714\]](#)

#### 6.3.1.15.1 Message Dashboard

With this app, you can get an overview of the interfaces you're responsible for and their current status. You can analyze the root causes of errors, restart or cancel data messages.

#### Key Features

You can use this app to:

- See all interfaces you've been assigned to using recipients.
- See the general status of your interfaces over time and restrict the time range.
- Check for active alerts and confirm them in the column *Alert*.
- Configure what notifications you want to receive for a specific interface in the column *Notification*. You can choose between *E-Mail for the Next Alert* (default), *E-Mail for Every Single Error*, and *No Mail*.
- See the message numbers and statuses (error, warning, success) by namespace and interface.
- Navigate to the message summary for an overview of how often a certain log message was written to the application log and how many data messages used a certain log message.
- Navigate to the monitoring and error handling to analyze the log messages in detail.
- Display the contents of the data messages.
- Create custom hints, message texts, and functions for certain log messages.

- Cancel data messages with issues that don't require you to take action (this functionality is only available for selected message formats).
- Solve errors by correcting the underlying data content of the messages.
- Restart corrected data messages or messages that had a temporary issue (this functionality is only available for selected message formats).
- Download payload for SOAP, IDoc, and XML messages.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

### 6.3.1.15.2 Message Monitoring Overview

With this app, you can see an overview of all the interfaces that you're responsible for monitoring. The intuitive card-based view provides a holistic overview of your message interfaces and also provides additional details like the number of messages in different statuses for each interface. When you click on the specific status for a specific interface, you're redirected to [Message Monitoring](#). Here, you can see a filtered view of the messages in the status that you selected for the specific interface containing the log information and key field information. For example, in the [Message Monitoring Overview](#) app, if you're monitoring an interface `AIF_XXX MMO` and in the card specific to `AIF_XXX MMO`, you click on the status `Error`, you're redirected to the [Message Monitoring](#) where you can see all the messages in status `Error` for the interface `AIF_XXX MMO`. You also see the log information and key field information for the messages in the `Error` status. This improves the efficiency of you finding the list of messages in a specific status for a given interface.

#### → Tip

If you don't see the interface you're assigned to, verify that your user is assigned to the corresponding recipient using the [Assign Recipients to Users](#) app.

## Key Features

This app provides the following key features:

- Monitoring overview of all the responsible interfaces

- Quick overview of key parameters of the selected messages
- Filter or search for messages with key content as the search parameters
- Navigate to *Message Monitoring* with appropriate filter conditions for specific interfaces and message status, significantly improving the user experience
- Order the cards to keep an eye on mission-critical interfaces using drag-and-drop mechanism
- Identify interfaces with an active alert

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

## Related Information

[Assign Recipients to Users \[page 2711\]](#)

### 6.3.1.15.3 Message Monitoring for Integration Experts

With this app, you can display an overview of your interfaces and get details on all messages processed through them.

#### → Tip

If you don't see the interface you're assigned to, verify that your user is assigned to the corresponding recipient using the [Assign Recipients to Users](#) app.

## Key Features

You can use this app to:

- Get a tabular overview of your interfaces and all messages (with specific status) processed through them.
- Filter your overview by time and/or interface, and restrict it to show only interfaces with messages or errors.

- Drill down on the messages of a specific interface and filter them by time, status, and key fields (if defined).
- Get more information on an individual message (such as log messages, message details, and data structure).
- Download the payload of a specific message.
- Edit, restart, or cancel a message (if in status *Error* or *In Process*) (this functionality is only available for selected message formats).
- Perform mass error handling for multiple messages of an interface at once.
- Set process information for a message (such as defining a process status).
- Identify interfaces with an active alert and confirm the alert.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

## Related Information

[Assign Recipients to Users \[page 2711\]](#)

### 6.3.1.15.4 Assign Recipients to Users

With this app you can assign your business users to the recipients that were defined for the communication scenarios. Once the users are assigned they can use the *Message Monitoring Overview* and *Message Dashboard* to monitor the interfaces and data messages they are responsible for.

## Key Features

You can use this app to:

- Find and add users.
- Find and assign pre-defined recipients to these users.
- Define which types of log messages the user can see in the Message Dashboard, for example, warnings only.

- Define if the messages are displayed or hidden on the Message Dashboard.
- Define if the user can see messages that are in process or that had a technical error using [Advanced Error Handling](#).
- Unassign recipients from users.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

### 6.3.1.15.5 Message Monitoring (Emergency Correction)

With this app, you can edit non-customized fields of a message payload in an emergency.

#### Key Features

You can use this app to:

- Get a tabular overview of your interfaces and all messages (with specific status) processed through them.
- Filter your overview by time and/or interface, and restrict it to show only interfaces with messages or errors.
- Drill down on the messages of a specific interface and filter them by time, status, and key fields (if any are defined).
- Get more information on an individual message, such as log messages, message details, and data structure.
- Download the payload of a specific message.
- Edit, restart, or cancel a message (if in status [Error](#) or [In Process](#)) (this functionality is only available for selected message formats).
- Set process information for a message (such as defining a process status).
- Identify interfaces with an active alert and confirm the alert.
- Edit the payload of messages, even if the messages aren't customized to be changeable.

Despite having similar functionalities as [Message Monitoring for Integration Experts](#), [Message Monitoring \(Emergency Correction\)](#) doesn't support navigating to the [Message Monitoring](#) app or performing mass error handling.

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

### **6.3.1.15.6 SOAP Error Log, OData Error Log, and Event Error Log**

With these apps, you can view and filter error logs related to SOAP web services, OData services, and events using SAP Application Interface Framework.

## **Key Features**

You can use these apps to:

- Get a comprehensive overview of failed communications relating to **SOAP web services**, **OData services**, and **events**
- Filter the SOAP Error Log, OData Error Log, and Event Error Log for various capabilities, such as Message ID, User Name, or Processing Date

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

## 6.3.1.15.7 Assigning the Business Catalog for SOAP Error Log, OData Error Log, and Event Error Log to Your Business Role

### Context

If you want to display the technical monitoring apps SOAP Error Log, OData Error Log, and Event Error Log in SAP Fiori launchpad, your user needs business roles that are authorized for the relevant business catalog.

### Procedure

1. In SAP Fiori launchpad, go to *Identity and Access Management* *Maintain Business Roles*
2. Select *Business Role*, then click *Edit*.
3. In the tab *Assigned Business Catalogs*, select *Add*.
4. In the pop-up *Assigned Business Catalogs*, select the following business catalog: *Communication Management – Technical Message Monitoring SAP\_CA\_BC\_COM\_TECH\_ERR\_PC*.
5. Click *OK*, then select *Save*. The tiles *SOAP Error Log*, *OData Error Log*, and *Event Error Log* are now visible in SAP Fiori launchpad.

## 6.3.1.15.8 Automatic Reprocessing

With this app you can configure the automatic reprocessing that is executed when a data message fails with a specific log message.

### Key Features

You can use this app to:

- Create, edit, and delete automatic reprocessing settings for a specific combination of log message and interface. Whenever a data message fails with the selected log message, the configuration you defined is executed.
- Define a maximum number of reprocessing attempts that can be executed for a failed data message.
- Define a minimum waiting time that must pass before the application can trigger a reprocessing attempt. The waiting time can take longer depending on the interval of the background job that executes the reprocessing.
- Define the intermediate status (In Process or Error) the failed data message displays while it's waiting to be reprocessed.

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-SRV-AIF.

### **6.3.1.16 Output Management**

[Maintain Print Queues \[page 2715\]](#)

[Monitor Email Transmissions \[page 2727\]](#)

#### **6.3.1.16.1 Maintain Print Queues**

##### **Purpose**

Using this app, you can set up print queues to manage the printing of documents and monitor the print jobs in each queue.

This helps you to identify and analyze errors and gives you a sense of direction for doing troubleshooting to solve them.

##### **Key Features**

With this app you can:

- Create a new print queue
- Delete a print queue
- Pause a print queue  
This allows you to stop print items from being collected by the printing process while you are carrying out repairs.
- Restart a print queue

- Check the status (such as *New*, *Failed*, *Transmitted*, *Successful*, *Warning*) of queues or individual print items in the queue and display detailed information regarding the status
- Reassign a print item to another print queue

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CCM-PRN-OM-PQ or BC-CCM-PRN-OM-PM.

## Related Information

[Install Additional Software \(SAP S/4HANA Cloud\)](#)

[Install Additional Software \(SAP BTP ABAP Environment\)](#)

### 6.3.1.16.1.1 Before Getting Started

#### Background Information for Print Queues

In a Cloud environment, the back-end system does not have a connection to local printers in your network (no virtual private network access is available, for example). To establish this connection, you need to create a print queue in the Cloud system representing an output channel to a local printer. To do so, open the *Install Additional Software* app and install the SAP Cloud Print Manager for Pull Integration (hereafter abbreviated as CPM) in your network. The CPM can then regularly check whether new print items are in the print queue. If this is the case, the CPM retrieves these items and sends them to the locally configured printer. In addition to that, you need to define communication scenario SAP\_COM\_0466. For more information on the CPM, see [3048273](#) or [Working with the SAP Cloud Print Manager for Pull Integration \[page 2718\]](#).

Watch the following video to learn how to get started with printing:

## Related Information

[SAP\\_COM\\_0466 \(SAP S/4HANA Cloud\)](#)

[SAP\\_COM\\_0466 \(SAP BTP ABAP Environment\)](#)

## 6.3.1.16.1.2 Creating Print Queues

You want to set up print queues to manage the printing of documents.

### Procedure

1. Open the *Maintain Print Queues* app.
2. Click *New* to create a new print queue.
3. Select a print manager.
4. Enter the required information:
  - **Queue:** Print queue name
  - **Description:** Print queue description
  - **Format:** The format is the printer language type of the queue. The following language types are available: PDF, CAB 203 dpi, CAB 300 dpi, Datamax 203dpi, Datamax 300 dpi, Datamax 406 dpi, Datamax 600 dpi, HP Laserjet 4350 PCL 5e, HP LaserJet 4350 PS, PCL 5c (Color), PCL 5e (Monochrome), Intermec 203 dpi, Intermec 300 dpi, Intermec 400 dpi, Lexmark T644 PCL 5e, Lexmark T644 PS, PostScript 2, Postscript 3, Toshiba 203 dpi, Toshiba 305 dpi, XMLDATA, Zebra 203 dpi, Zebra 300 dpi, Zebra 600 dpi.
  - **Communication User:** Technical user with which the *SAP Cloud Print Manager for Pull Integration* logs on to the system. You have to define a communication user in the communication scenario SAP\_COM\_0466 first. You can connect several queues to the same *SAP Cloud Print Manager for Pull Integration* - in this case all queues must use the same communication user.
  - **Retention:** Spool Retention Period. The retention period is the number of days after which the print queue items that are in final state (*successful* or *failed*) will be deleted automatically.
5. Click *Create* to create a new print queue.

### Results

You have created a print queue you can use in the *SAP Cloud Print Manager for Pull Integration*.

#### ⓘ Note

The *Maintain Print Queues* app always includes a print queue called **DEFAULT**. It's the only print queue provided by SAP and serves as a sample queue that can't be used for productive output as it can't be connected to the *SAP Cloud Print Manager for Pull Integration*.

To connect to your physical printers, you therefore need to create one or more custom print queues.

Keep in mind that you can't define custom retention time for items in the **DEFAULT** queue and that SAP might adapt the standard retention time at any time to avoid misuse. The **DEFAULT** print queue should also not be used as storage for all documents that are printed from the system, as SAP might restrict the given quota at any time to avoid misuse. A cleanup job will delete all items in the **DEFAULT** print queue after eight days, regardless of their status. This will not affect the behavior of other print queues.

## Related Information

[SAP\\_COM\\_0466 \(SAP S/4HANA Cloud\)](#)  
[SAP\\_COM\\_0466 \(SAP BTP ABAP Environment\)](#)

### 6.3.1.16.1.3 Working with the SAP Cloud Print Manager for Pull Integration

You can use the SAP Cloud Print Manager for Pull Integration (from now on abbreviated as CPM) to establish a connection between an SAP Cloud-based solution and printers available in your network. It requires the CPM to enable automatic transmission and printing of documents from the cloud-based system to local printers.

You should install the CPM once in your network. Download the CPM installation package from the [Install Additional Software](#) app and select *Run as Administrator*.

#### ⓘ Note

Keep in mind that you need to have administrator rights in order to proceed with the installation. The installation should be performed on a Windows server.

Start the downloaded installer executable file. The SAP Front-End Installer opens. Select CPM and click *Next*. Wait until the installation has finished and confirm the final dialog. The CPM is now installed.

As a Windows® service, CPM is available for print queue output from the cloud-based system as long the Windows® computer is up and running and the CPM Windows® service has not been stopped manually. Thus, we recommend to install CPM on a central print server that is up and running all the time or at least during the time in which printing is needed.

You only need to install CPM more than once if you have multiple separate networks with printers that you want to use for process-integrated printing.

## Prerequisites

- **Prerequisites on the Print Server**

You have administration authorization for the computer on which you install CPM.

**Note that** when CPM collects documents from a print queue and sends them to a printer, the documents are stored temporarily on the server CPM is installed on. If you are using CPM to print sensitive documents, it is recommended that you ensure that only authorized users can access this folder. The folder, called the working directory, can be found in the following path on the Windows® computer running CPM:

C:\ProgramData\SAP\Cloud Print Manager for PI

Since the CPM can run under a different user than the SAP Cloud Print Service, the working directory is stored under the ProgramData folder as it can be accessed from both users.

For the storage of the temporary documents and the configuration, the embedded NoSQL database LiteDB is used. LiteDB is a serverless database fully written in .NET C# managed code. The stored data can be viewed with the [LiteDB.Studio](#) .

- **Prerequisites in the SAP Cloud Systems**

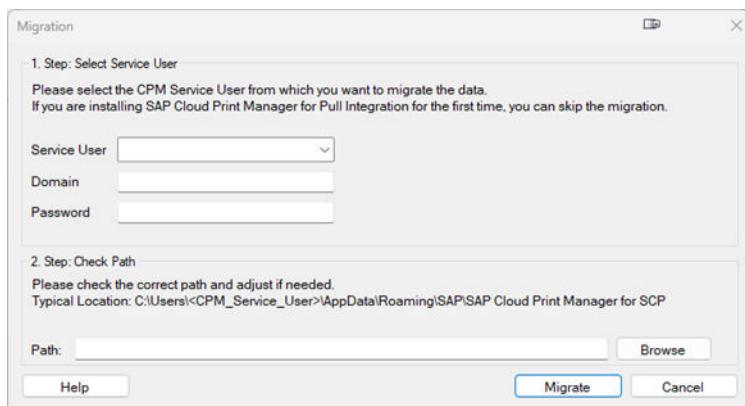
In S/4HANA Cloud the integration with CPM is done with an appropriate communication scenario. Please follow the instructions in [Integrating SAP S/4HANA Cloud and Local Printers](#). For SAP Business Technology Platform please get more information from the SAP Print Service documentation on [how to integrate your system with SAP Cloud Print Manager for Pull Integration](#).

## Migrate to Release 800

Starting with release 800 of the CPM some major changes have been made that require a migration. For this, a new migration tool was introduced. At the first start the migration tool is launched automatically.

### ⓘ Note

Please make sure that the print service is stopped, and that the migration process is started during a phase with low printing amount.



1. Here you must select the CPM service user that you have previously used and from which you want to migrate the data. It is also required to type in the corresponding password to perform the migration.
2. Please verify that the path to the CPM application data is filled and correct. Usually, nothing needs to be changed here.

After the migration is done, the service user selected in the migration tool is automatically set as CPM service user as well. The service user can be changed as described here: [Setting Up the SAP Cloud Print Manager for Pull Integration \[page 2719\]](#).

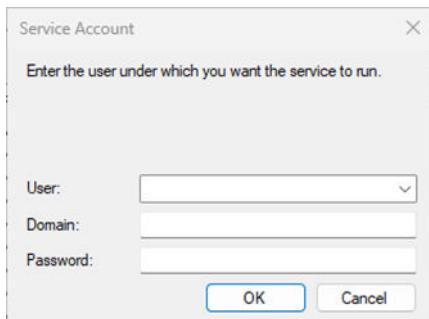
### 6.3.1.16.1.3.1 Setting Up the SAP Cloud Print Manager for Pull Integration

In the following subsections, you find a description of the different steps necessary to set up the CPM.

## Define Connection Settings

1. Start the CPM application from the Microsoft Windows® start menu.
2. Set or change the service user account (optional step) (This step is only required if the service user was not already set during the migration process.)

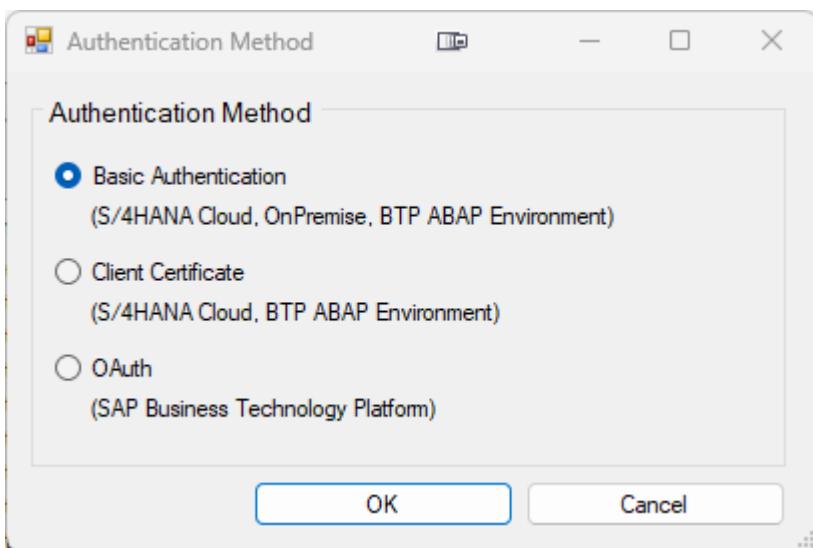
At the first start of the CPM, the service account dialog opens automatically. In case the dialog doesn't open, or you want to change the service user, you can select *Service User* from the *Settings* menu. Enter a service user to run the Microsoft Windows® service. This user should have administration rights on the local computer as well as authorization to access your network printers. The user dropdown shows only local user accounts, but you can also use a domain user by simply filling in the required fields. We recommend to use a local user account as service user.



3. **Optional step:** From the settings menu, select *Proxy* to set up a connection to the internet. It is recommended to use the same configuration required by your company to access the internet. Mostly, the default setting is sufficient and there is no need to change anything.  
After applying these settings, the print service is started and runs as the Windows® service CPM. In addition, as a second Windows® service, you can start the CPM to check if the print service is running. If it's not running, for example, due to a crash, the CPM restarts the print service to ensure permanent availability of the service. If necessary, you can restart the print service if it has stopped for any reason. You can start both services in the top right corner of the screen.

## Add a New Runtime System

From the runtime system menu, select *New* to add a new runtime system to the CPM.



## Basic Authentication

This method can be used for S/4HANA Cloud systems, OnPremise or BTP ABAP Environment systems.

1. Select *Basic Authentication* and confirm the dialog.
2. Enter the following information:
  - Name: Enter a unique name for the system.
  - SAP Web Service URL: Enter the URL to your system provided by SAP.
  - User: Enter the communication user you created while setting up the communication scenario for CPM.
  - Password: Enter the communication user's password.
3. Click *Test* to test the connection to the system and the user credentials.
4. Click *OK* to save the new configuration.

## Client Certificate

This method can be used for S/4HANA Cloud systems or BTP ABAP Environment systems.

1. Select *Client Certificate* and confirm the dialog.
2. Enter the following information:
  - Name: Enter a unique name for the system.
  - SAP Web Service URL: Enter the URL to your system provided by SAP.
  - Certificate: Select the client certificate that should be used to establish the connection. The dropdown list displays all the certificates installed in the *Personal* folder of the Windows® machine certificate store. The client certificate must be signed by an appropriate certification authority (CA) and uploaded to the communication user in the S/4HANA Cloud system. A list of all root CAs approved by SAP Global Security is available in SAP Note [2801396](#).
3. Click *Test* to test the connection to the system and the user credentials.
4. Click *OK* to save the new configuration.

## OAuth

This method can be used for SAP Business Technology Platform.

1. Select *OAuth* (SAP Business Technology Platform) and confirm the dialog.

2. Enter the following information:
  - Name: Enter a unique name for the system.
  - SAP Web Service URL: Enter the URL to your system provided by SAP.
  - Client ID: Enter the client ID you received during the registration.
  - Client Secret: Enter the client secret you received during the registration.
  - Token URL: Enter the token URL you received during the registration.
3. Click *Test* to test the connection to the system and the user credentials.
4. Click *OK* to save the new configuration.

As soon as you have added the runtime system, CPM starts to collect print queues from the SAP Cloud system. All print queues that have been assigned to the communication scenario or user are displayed in CPM. You now need to assign Windows® printers to the print queues to start printing. If you don't immediately see any print queues, select *Check Now* from the *Runtime System* menu to start the connection.

## Remarks

- It is not supported to connect to the same runtime system with the same communication user multiple times.

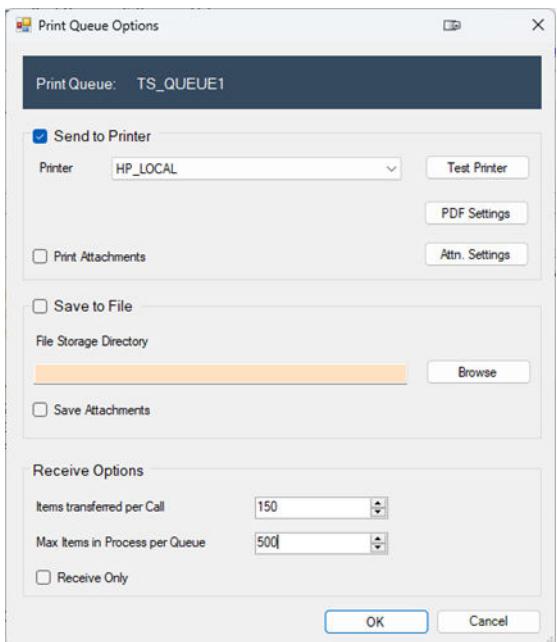
## Assign a Printer to a Print Queue

1. Under *Runtime Systems*, select a system from the list. All print queues that have been assigned to the technical user are displayed in this list.
2. Under *Print Queues*, select a print queue and choose *Queue Options* from the *Print Queue* menu or use the context menu or a double click on the queue. The *Print Queue Options* dialog opens. Mark the checkbox *Send to Printer* and save the file.
3. Select a Windows® printer and browse or enter a path and click *OK* to save the configuration.

Note that the printer that you assign to the queue should support the printer language defined in the queue. You can view this information in the *Format* column of the *Print Queues* list. Do not select a printer that requires user interaction for each print job. For example, you should not select a printer that prints to a file with a file selection popup. There is no user interface to the service, so the job wouldn't print.

As soon you have assigned a printer to the print queue, the documents that have been sent to this print queue will be collected and printed.

CPM can print or save attachments which were added to the document. Attachments can be saved in any format, but you can only print PDF, DOCx, XLSx and PPTx attachments. To save attachments to the specified path, select the checkbox *Save Attachments*.



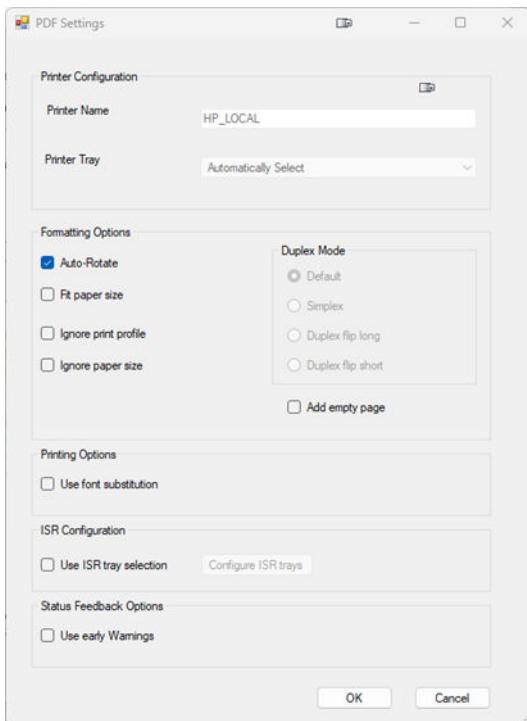
### ⓘ Note

Mapped network drives might not work correctly if chosen from the *Browse* dialog. In this case, enter the UNC path name (`\\\<directory_name>`) manually in the entry field for the file storage directory on the left side.

## Test Printer

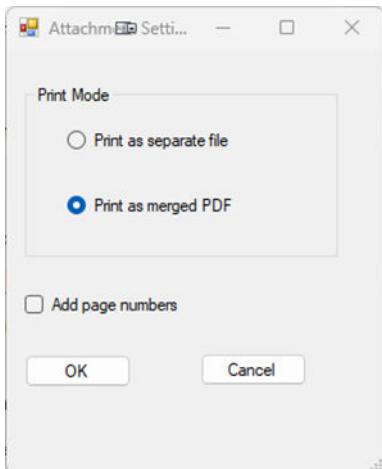
1. Under *Print Queues*, select a *Print Queue* and choose *Output Options* from the print queue menu.
2. Ensure that *Send to Printer* is checked and a valid printer is selected.
3. Press the *Test Printer* button. A test page is sent to the defined printer. The test page contains a test sentence in English, Chinese, French, and German.

## PDF Settings



1. Click on *PDF Settings* to define print settings for your PDF document. This button is only available for queues with a PDF format.
2. Choose between the following options:
  - Auto-Rotate: the print settings are adapted in accordance with the orientation of the document that you want to print.
  - Fit paper size: the print settings are scaled in accordance with the paper size in the paper tray while preserving the aspect ratio of your original document.
  - Ignore print profile: CPM sends printer specific properties of the attached printer to the print queue in the SAP Cloud system. If supported from the application, these can be used to define print profiles for a print queue item. Print profiles can consist of settings like duplex mode, color mode, tray selection and others. To use the global default settings of CPM, select this option.
  - Ignore paper size: sending a document with invalid paper size (for instance, a letter to an A4 tray) usually causes the printer to ask for a manual confirmation to print. To prevent this, select this option.
  - Duplex Mode: global print property (see 'Ignore print profile')
  - Add empty page: When selecting this option, an empty page is added after a document with odd pages is printed in duplex mode.
  - Use font substitution: see SAP note [2959085](#) for details.
  - Use ISR tray selection: configure swiss payment form printing. Do not use this for any other printing.
  - Use early warnings: sends early status feedback to the S/4HANA Cloud or SAP BTP print queue in case of a printer issue. For example: If the printer is out of paper, all items will be set to warning.
3. Confirm with *OK*.

## Attachment Settings



1. Click on [Attachment Settings](#) to define print settings for your attachments.
2. Choose between the following options from the print mode:
  - Print as separate file: All attachments are printed separately keeping their original document format.
  - Print as merged PDF: All attachments are merged into one PDF and are printed in accordance with the selected PDF settings (see section [PDF Settings](#)).
3. Additionally, you can select [Add Page Numbers](#) for merged PDFs to add page numbering.
4. Confirm with [OK](#).

## Receive Options

Below, you'll find a list of all available Receive options:

- [Items transferred per Call](#) specifies the number of print queue items which are transferred in one request using the new print API.
- [Max. Items in Process per Queue](#) specifies the maximum number of print queue items that can exist in the internal queue of CPM. If the limit is reached, no new items will be received until some of them were processed. This behavior should prevent memory overflow in case of problems with a printer.
- [Receive only](#) is a test option to measure the performance of the item transfer from the SAP Cloud system to CPM. Do not select this option in a productive environment. Nothing is printed or saved. Create many items in a test queue, set the [Log Level](#) to [Info](#) (see section [Other Functions](#)) and select [Receive only](#). After all print queue items were received, you can find a time stamp for each block of items in the [Error Log](#). The block size is the number specified in [Items transferred per Call](#).

## Other Functions

### View Log Information

1. From the Help menu, select [About](#).

2. Using the links provided in the *About* CPM dialog, you can access the following information:
  - Document Log: All documents retrieved from the configured runtime system.
  - Error Log: General errors.
  - Printing Log: Errors during print job processing.
3. If you need to provide more detailed log information, select *Options...* from the *Settings* menu, set the *Log Level* to *Info* or *Debug* and mark the checkbox named *Keep Job-Specific Log Files*. Then reproduce the error. This will add more detailed error information to the log files.
4. If required, click *Save Support Information* to create an archive file containing all log information, which can be attached to a support incident on the application component BC-CCM-PRN-OM-PM.

## Cleanup Directories

1. From the settings menu, select *Clean up Directories*.
2. All documents and log files stored in CPM's working directories will be deleted.

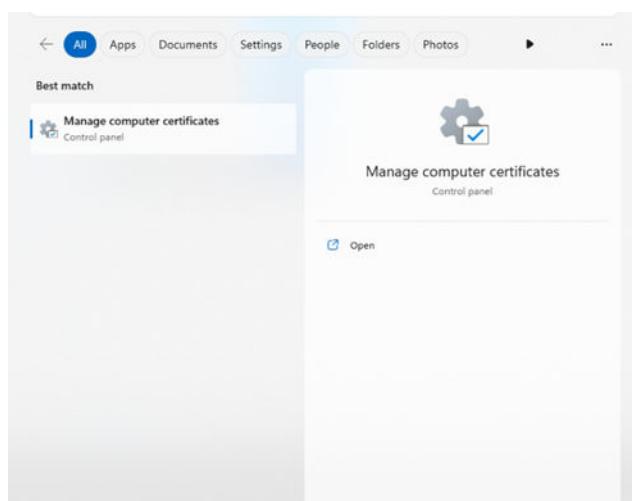
## Reset to Factory Defaults

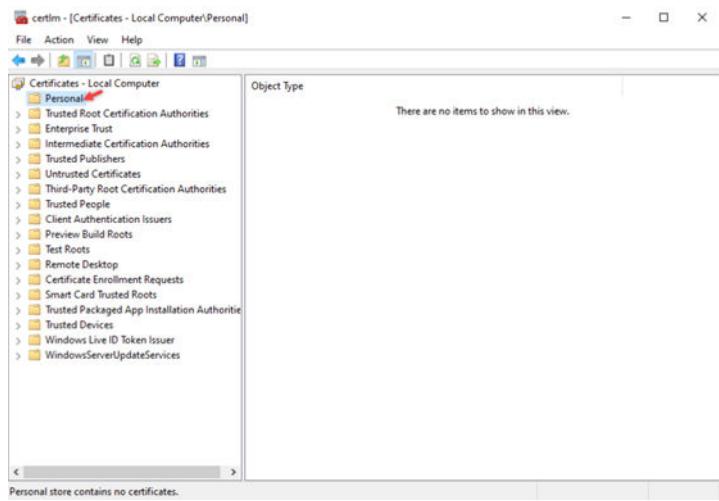
1. From the *Settings* menu, select *Reset to factory defaults*.
2. All documents and log files, including the configuration, will be deleted. After performing this action, all the system information, including user and password, must be entered again.

## Appendix

### Self-signed Certificate

To use self-signed certificates, they must be installed in the Windows® computer Certificate Store under the *Personal* folder. After the certificate is installed successfully, the CPM service must be restarted. You can then add connections to the system using the self-signed certificate.





### 6.3.1.16.2 Monitor Email Transmissions

#### Use

You can use this app to display successful and failed email transmissions and to provide background information. This app is read-only and doesn't allow any data changes.

#### Key Features

With this app, you can:

- Search for a specific email by adapting filters
- Sort, add, or remove columns
- Display all outgoing email requests
- Export a results list as a spreadsheet
- Get the header data and information about the sending status on a specific email request
- Send a test email to check if the email outbound in a system is working

#### Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-COM.

## Related Information

[Monitoring Email Transmissions \[page 2728\]](#)

### 6.3.1.16.2.1 Monitoring Email Transmissions

Find out how to monitor your email transmissions.

## Procedure

Start the [Monitor Email Transmissions](#) app. On the starting page, you can search for a specific email by adapting the following filters:

- Subject
- Status
- Sender
- Created at
- Recipient Address

In addition, you can

- Sort columns
- Add or remove columns

Click [Go](#) to display the search results. In the results list, all email requests are displayed according to the filters you've set. You can export the results list as a spreadsheet.

To get more information on a specific email request, click a request in the results list. On the [Details](#) page, the following information is displayed:

- Status
- Sender
- Status Code
- Subject
- Created at
- Response

You can also get information about the recipients, such as:

- Latest status

- Recipient address
- Copy (BCC or TO)
- Latest status changed at

You can get additional information about the present and former recipient's status by selecting a recipient.

## Email Reorganization

Due to data protection reasons, all displayed emails will be deleted from the table view after 30 days.

## Testmail Functionality

You can send out a test email to any recipient chosen by you. The subject and the text are fixed values that are set by the app. The mail sender will automatically be set as the email address of the user currently logged on, or, if that user has no email address, as the default email address. After the successful creation of the send requests, the email can be viewed in the main table of the *Monitor Email Transmissions* app.

Sending test emails can help administrators to check if the email outbound of their system is working. If the email outbound is not working, you'll either directly receive an error message, or you can see the error message in the main view of the *Monitor Email Transmissions* app.

## Related Information

[Monitor Email Transmissions \[page 2727\]](#)

[Configuration of the System Email Outbound](#)

### 6.3.1.17 Read Access Logging

#### Definition

Read Access Logging (RAL) is used to monitor and log read access to sensitive data. This data may be categorized as sensitive by law, by external company policy, or by internal company policy. The following typical questions might be of interest for an application that uses Read Access Logging:

- Who accessed the data of a given business entity, for example a bank account?
- Who accessed personal data, for example of a business partner?
- Which employee accessed personal information, for example religion?

- Did anyone search, for example, if VIPs were admitted to hospital?
- Which accounts or business partners were accessed by which users?

These questions can be answered using information about who accessed particular data within a specified time frame. Technically, this means that all remote API and UI infrastructures (that access the data) must be enabled for logging. Read Access Logging is available for different channels such as Dynpro and Web Dynpro.

### Main Purposes of Read Access Logging

Read Access Logging is often required to comply with legal regulations or public standards such as data protection and privacy, for example in banking or healthcare applications. Data protection and privacy is about protecting and restricting access to personal data. In some countries, data protection and privacy regulations even require that access to certain personal data is reported. Companies and public institutions may also want to monitor access to classified or other sensitive data for their own reasons. If no trace or log is kept on who accesses data, it is difficult to track the person responsible for any data leaks to the outside world. Read Access Logging provides this information.

Read Access Logging is always based on a logging purpose that is freely defined according to the requirements of an organization (for example, data protection and privacy). This logging purpose is then assigned to each log entry as an attribute, which allows the log data to be classified and organized according to the logging purpose. For example, various archiving rules or reportings can be created based on logging purposes.

The Read Access Logging framework can thus be used to fulfill legal or other regulations, to detect fraud or data theft, for auditing purposes, or for any other internal purpose.

### Background Information on the Architecture of Read Access Logging

When an application is started, the Read Access Logging configuration is read. It indicates whether, for example, the current function module, operation, or UI element is log-relevant and to what extent. For example, should only the access itself be logged, or the content too? Log entries can be structured according to their semantics.

SAP applications may contain predefined configurations. However, the customer's key user will usually have to adapt them to his or her needs in order to fulfill the legal requirements of his or her organization - requirements that are not all known to SAP.

#### Note

- As with all logging, your need to log data must be balanced with the effect logging will have on the performance of your system. The performance of your system will depend upon the amount of data you log, as well as the complexity of the conditions you specify for which data is logged.
- You can find information on Read Access Logging configurations and templates in specific areas in the documentation for those areas.

## Activities

### Configuration Tasks

1. Determine which data must be logged under which circumstances.

The organization must define which legal or security requirements to apply and which data must be logged on a semantic level. For example, the legal department might determine that access to the social security number must be logged.

2. Define purposes for Read Access Logging.  
Depending on the logging purpose and the laws and regulations that must be fulfilled, the application can define reports and rules for saving and processing the data. For more information, see [How to Define Purposes of Read Access Logging \[page 2734\]](#).
3. Determine the channels (such as Dynpro or Web Dynpro) through which the data can be accessed.
4. Define log domains.  
Log domains are groups of semantically related data fields that need to be logged. For example, *Gross salary* and *Net salary* might be grouped together in the *Salary* log domain. Log domains bundle different technical representations of the same semantic entity. Log domains are channel-independent. For more information, see [How to Define Semantic Grouping of Fields to Be Logged \[page 2735\]](#).
5. Define which data needs to be logged and whether only the access is logged or also the content.  
For more information, see [How to Define What to Log \[page 2738\]](#).

## Tasks During Operations

- Defining a user exclusion list for Read Access Logging configurations.  
If you want to exclude certain users from Read Access Logging, they need to be added to a user exclusion list. This is useful for any automatic processing without user interaction, for example, background jobs. For more information, see [How to Exclude Specific Users from Read Access Logging \[page 2754\]](#).
- Displaying changes made to Read Access Logging configuration and evaluating errors and warnings.  
For more information, see [How to Check Runtime Errors and See Changes to Configurations \[page 2754\]](#)

## Monitoring

You can view all log entries using the Read Access Logging monitor. For more information, see [How to Monitor the Read Access Log \[page 2757\]](#).

## Related Information

[SAP Signavio Process Navigator for Data Protection and Privacy \(Scope Item 5LE\)](#) 

### 6.3.1.17.1 Read Access Logging for Data Protection

You can use Read Access Logging (RAL) to monitor and log access to personal data. The information provided may include, for example, which business users accessed business partner personal data, and in which time frame.

SAP delivers default configurations, which assign dedicated fields to log domains. A log domain is a category that groups semantically identical or related data fields. Logging occurs for all fields disclosed on the UI that are related to these domains. The domain is displayed in the log.

You can activate or deactivate the available RAL configurations in the *Read Access Logging Configuration* app, or make changes. You should carefully consider which information is relevant for logging. Configure your system to log only what you really control. If you maintain a wide scope of information to be logged, you will end up with a lot of data that will be more difficult to process than if you are more specific in your logging configurations.

The following table lists the log domains that are part of the delivered sample configuration content. You can use these log domains or define your own, according to your needs.

Log Domain Name	Description
BANK	Data referring to a bank account
BIOMETRIC*	Data referring to biometric data
CRIME*	Data referring to criminal or administrative offenses or suspected criminal or administrative offenses
ETHNIC_ORIGIN*	Data referring to racial or ethnic origin
GENETIC*	Data referring to genetic data
HEALTH*	Data referring to health data
POLITICAL_OPINION*	Data referring to political opinion
PROFILE*	Data which is usually based on profiling like: scoring, rating, unwanted customer
RELIGION*	Data referring to religious or philosophical beliefs
SECRECY*	Data referring to professional secrecy
SEX_LIFE*	Data referring to sex life
SEXUAL_ORIENTATION*	Data referring to sexual orientation
SSN	Data referring to social security number
TRADE_UNION*	Data referring to trade union membership

These log domains are logged with fields related to business partner (**BP**), customer (**CUSTOMER**), supplier (**VENDOR**), legal entity (**LEGAL\_ENTITY**), employee (**EMPLOYEE**), or student (**STUDENT**) because they are only identifiable with this additional personal data. For example, **TRADE\_UNION** data requires details on the **EMPLOYEE**.

Log conditions are used, if required, to limit the logging of data (in technical terms, these fields are considered as **and** conditions). For example, you could configure RAL to log a field related to the employee in a specific transaction only if the employee's religion is shown. If this field is only visible on a tab where the employee's religion is not displayed, access to this field is not logged.

RAL is switched off by default. You can activate it in the respective section of the [Read Access Logging Configuration](#) app. In the [Read Access Logging: Monitor](#) app, you can view created logs. If you need a downloaded version of your RAL logs, please contact the Service Center. Also contact the Service Center if you need to create new configurations.

## More Information

For more information on Read Access Logging and the related apps, see [Read Access Logging \[page 2729\]](#).

### 6.3.1.17.2 Read Access Logging Configuration

With this app you can configure Read Access Logging to determine which read access to data is logged and under which conditions.

#### ⚠ Restriction

RAL items can not be created in the P system, only in the Q system. If you try to create configurations, log domains, purposes or recordings in the P system, you will get an error message saying `Items of type SRAL_<RAL Item> must not be edited in this system`. You can still activate and deactivate RAL items in the P system.

## Key Features

You can use this app to:

- Define purposes of Read Access Logging
- Define semantic grouping of fields to be logged
- Manage recordings of application user interfaces
- Change predelivered configurations of what to log
- Enable Read Access Logging in current client
- Check runtime errors and see changes to configurations

## Prerequisites

To access the app, you need to have the `SAP_CORE_BC_RAL_CONF` business catalog assigned to your user. This business catalog is contained in the `SAP_BR_ADMINISTRATOR` business role template.

## Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SEC-RAL.

### 6.3.1.17.2.1 How to Define Purposes of Read Access Logging

Read Access Logging is always based on a logging purpose that is freely defined according to the requirements of an organization. It describes why specific data is logged.

#### Context

In the configuration, you specify the logging purpose and each log entry in the log is assigned its purpose as an attribute. This configuration enables you to organize the log data by the logging purpose. For example, various archiving rules or reportings can be created based on logging purposes.

In each Read Access Logging configuration, assign a logging purpose. When you define archiving rules, use the logging purpose as the basis.

#### Procedure

1. In the *Read Access Logging Configuration* app, choose *Logging Purposes*.
2. Decide whether to create a purpose or modify one of the templates delivered by SAP.

SAP delivers default purposes you can copy and modify to your needs or you can create your own.

- To create your own purpose, choose *Create* under *Search Results*.
- To copy a purpose from SAP, do the following.
  - a. Under *Search Criteria*, choose *Search*.
  - b. From the search results, select a template from SAP.

Templates from SAP use the  (*SAP Template*) icon under *Owner* or list SAP under *Created By* or *Changed By*.

- c. Under *Actions*, choose  (*Copy*).

3. Enter the required data.

The ID is limited to ten characters. We recommend that you use an abbreviation of the purpose name as ID.

The purpose name appears on all UIs.

4. Save your entries.

## Results

Assign logging purposes during the creation of Read Access Logging configurations. Logging purposes help you with the definition of archiving rules.

### 6.3.1.17.2.2 How to Define Semantic Grouping of Fields to Be Logged

Within an application, the data to be logged must be defined on a semantic level, before the actual fields and rules are defined. Log domains are semantic descriptions of semantically identical or related fields that have different technical representations.

## Context

In Read Access Logging, you define a log domain. When changing Read Access Logging configurations, assign a log domain to each field to be logged.

For a log domain, specify a name and a business area that the data element is related to. This definition is necessary because different applications might use the same log domain. For example, a log domain *account* can be something different in the Human Resources application than it is in the Banking application.

## Procedure

1. In the *Read Access Logging Configuration* app, choose *Log Domains*.
2. Decide whether to create a log domain or modify one of the templates delivered by SAP.

SAP delivers default log domains you can copy and modify to your needs or you can create your own.

  - To create your own log domain, choose *Create* under *Search Results*.
  - To copy a log domain from SAP, do the following.
    - a. Under *Search Criteria*, choose *Search*.
    - b. From the search results, select a template from SAP.
3. Enter the required data.

You can freely define the business area. The business area functions as a namespace for the data element. The description appears in the detailed view of the read access log and can be helpful for the person evaluating the log to identify the log domain.
4. Save your entries.

## Results

When you change a configuration, you can assign a log domain to each field to be logged. The log domain is displayed in the read access log and helps the log evaluator to understand the semantics of the field. It enables the evaluator to search for a given semantic field regardless of the technical representation or ID of this field.

### 6.3.1.17.2.3 How to Manage Recordings of Application User Interfaces

To use Read Access Logging with user interface technologies like Web Dynpro and Dynpro, you first identify the log-relevant fields. Read Access Logging provides a user interface recorder to identify those fields.

## Context

When read access to remote APIs is logged, their contents are already known at design time. All fields of a Web service are therefore available for read access logging configuration. For user interface technologies however, the elements that are visible are not known until the application is started. Therefore, it is not possible to offer all fields for read access logging configuration. First, determine which fields are needed for read access configuration. A user interface recorder, which is started from the [Read Access Logging Configuration](#) app, helps you. When the recorder runs, the key user navigates the user interface and records every UI element that is relevant for read access logging.

#### ⓘ Note

The value of the fields that are displayed on the UI during the recording (the so-called sample values) become visible in the recording. The sample values and UI labels help the key user to identify the fields when changing a configuration. However, you can rerecord or delete the sample values. For example, if you find it more helpful for the key user, you can replace them with a description of the field. You can also search for one or more recordings and clear all sample values at once. Select one or more recordings from the search results and choose [Clear Sample Values](#).

#### → Tip

During the recording, the field labels are recorded in the UI language in which the recording is made. These labels are displayed in the configuration using the same language of the user interface recording. We recommend that you perform the recording in a language that the key user, who creates the configuration, understands.

## Procedure

1. In the [Read Access Logging Configuration](#) app, choose [Recordings](#).

2. Decide whether to create a recording or modify one of the default recordings delivered by SAP.

SAP delivers default recordings you can copy and modify to your needs or you can create your own.

- To create your own recording, choose *Create* under *Search Results*.
- To copy a recording from SAP, do the following.
  - a. Under *Search Criteria*, choose *Search*.
  - b. From the search results, select a template from SAP.

Templates from SAP use the  (*SAP Template*) icon under *Owner*.

- c. Under *Actions*, choose  (*Copy*).

3. Enter the required data.

4. Save your entries.

- If you copied a recording, you can use the copy as is or modify the recording to fit your needs. You can skip the rest of the steps in this procedure.
- If you created your own recording, the recording is created and started. You can now start the user interface you want to record. The recorder stays in the *Recording* state until you stop it (choosing the *Stop Recording* icon). You can restart the recording at a later point in time by choosing the *Start Recording* icon.

Perform the steps as follows.

5. Start the application that you want to record.

 Note

If the application is already open when you create the recording, you must restart it. The check if Read Access Logging is activated or not is performed when an application is started.

6. Right-click (Web Dynpro) or `Ctrl` + right-click (Dynpro) each field that you want to collect for the recording and choose  *Read Access Logging*  from the context menu.
7. To remove a field from a recording, right-click the field and choose  *Read Access Logging*  *Remove Field from Recording* .
8. When you have recorded all fields, close the application and return to the Read Access Logging Configuration app and choose *Stop Recording* in the list of recordings.

If you do not close the Web Dynpro session, the recording is locked and you cannot change it.

## Results

A recording for a Dynpro or Web Dynpro application is created. If the recording includes errors or is missing values, restart the recording.

 Tip

When you restart a recording, you can select another user to be recorded. This function enables you to have another user carry out the steps you want to record with their authorizations in the system. You avoid running the recording with missing or too many authorizations.

You can also download and upload the recording.

### Note

If you record fields from different applications (Web Dynpro) or programs (Dynpro) within one recording, the recording is separated into the programs and applications that were used. Create configurations for each program or application within the recording.

For Dynpro, single transactions can contain more than one program, which may not be apparent to you when you create the recording. In this case too, your recording is separated into the programs that were used.

## 6.3.1.17.2.4 How to Define What to Log

To define what to log, use a read access logging configuration.

### Context

There are pre-existing configurations delivered by SAP that you can use immediately. If you need a configuration that is more customized to your needs, you can either create a configuration from scratch or change an existing one.

The configuration can then be activated and used for different applications.

### Related Information

[How to Create a New Configuration \[page 2738\]](#)

[How to Activate or Deactivate the Correct Read Access Logging Configuration \[page 2740\]](#)

## 6.3.1.17.2.4.1 How to Create a New Configuration

You can define what to log by creating your own read access logging configuration or modifying a template delivered by SAP.

### Procedure

1. In the [Read Access Logging Configuration](#) app, choose *Configuration*.
2. Select a channel.
3. Decide whether to create a configuration or modify one of the templates delivered by SAP.

SAP delivers default configurations you can copy and modify to your needs or you can create your own.

- To create your own configuration, select a channel and choose *Create* under *Search Results*.  
For channels that use recordings, choose a recording.

- To copy a configuration from SAP, do the following.
  - a. Under *Search Criteria*, filter with **Owner is SAP Template**.
  - a. From the search results, select a template from SAP.

Templates from SAP use the  (SAP) icon under *Owner*.

- b. Under *Actions*, choose  (Copy).

For channels that use recordings, copy the recording that comes with the template under the same name or enter a new name.

- c. Find the channel you just copied and choose  (Edit Configuration).

4. Depending on the channel, select the object you want to create a configuration for.

For example, an operation for a Web Service, a function module for an RFC, or an application for a Web Dynpro. If you are editing an existing channel, the object is already selected for you.

For more information, see [Channel-Specific Information \[page 2742\]](#).

5. Enter the required data.

- For Dynpro, choose *Application / Software Component*.
- For Dynpro and Web Dynpro, specify a *Log Context*.

6. Create log groups.

#### Note

For a logging purpose, only create one logging group.

- a. Under *Log Groups*, choose  (Create Log Group).
- b. Specify a logging purpose and a description for the log group.
- c. Specify the fields that you want to log together in the log group.

Move fields in the *Field list* to the log group.

- d. In the *Fields* list, define how each field should be logged.
  - The *Logging Type* determines if only the field name is logged when accessed or if the value of that field is also logged.
  - To save space in the database, set *Exclude if Initial* to have Read Access Logging ignore a field if its value is initial.

#### Note

The initial value is not the default value, but the technical initial value. For example, the empty value for string or character-based fields or the number zero for integers. The default value is logged in any case.

If this option is set and all fields within a Web service, RFC, or Web Dynpro UI are initial, no log entry is created for any of the fields.

- For Dynpro and Web Dynpro, specify the *Field Type*, which is whether an input field, an output (display) field, or both.

### Note

Values for password fields cannot be included in logs. The *Logging Type* is *Without Value* and cannot be changed.

## 7. Create conditions.

- Under *Conditions* choose  (*Create Condition*).

If you do not assign any conditions, select the *Without Condition* checkbox in the log group attributes and continue with the next step.

- Specify a name and a description for the condition.
- Create one or more expressions for the condition.

Expressions are joined using logical AND.

- Specify the fields that you want to define rules for.

Move fields in the *Field list* to the expression.

- For each field, specify the select option (the selection criteria).

For example, *Username Inclusive Equals John Smith*.

Select options are joined using logical OR.

- Assign conditions to a log group or set the *Without Condition* option.

To assign a condition, select the Log Group, choose *Assign Condition* and select a condition. Activate the assignment if you want to apply the condition to the log group. The log groups and conditions do not need to be activated, however.

## 8. To check the configuration for consistency and function: choose (*Check Configuration*) Check.

## 9. Save your entries.

### 6.3.1.17.2.4.2 How to Activate or Deactivate the Correct Read Access Logging Configuration

#### Context

To activate or deactivate Read Access Logging for certain apps, follow the steps described below.

#### Procedure

- Go to the app for which you want to activate or deactivate Read Access Logging.
- Open the app and navigate to your user area (*User Actions Menu*).
- Click the *About* icon.

- If the app has an app ID such as F1492, please proceed with the steps in the section *Activating and Deactivating Read Access Logging for Apps with App ID*.

#### **Activating and Deactivating Read Access Logging for Apps with App ID:**

4. Copy the app ID.
5. Go to the [SAP Fiori apps reference library](#) and select *All Apps*.
6. In the *Search* bar, paste the app ID you have just copied.
7. Click the app entry in the results list.
8. On the *App Details* screen, select *SAP S/4HANA Cloud* from the dropdown list.
9. In the section *Implementation Information*, select your current SAP S/4HANA Cloud release.
10. Open the subsection *Configuration* and copy the OData service name/s and the OData service version of the app.
11. Go back to the SAP Fiori launchpad and open the app Read Access Logging Configuration.
12. Go to *Configuration*.
13. Select the channel *SAP Gateway*.
14. Paste the OData service name that you have just copied in the field *Service ID*.

If you have copied more than one OData service name, you can add further *Service ID* fields to your search criteria.

15. Paste the OData service version that you have just copied in the field *Service Version*.
16. Click *Search*.

In the search results list, you can see if Read Access Logging configurations exist for the service/s and, thus, for the app, and if the Read Access Logging is active or inactive.

17. To activate or deactivate Read Access Logging, select the service/s, and click either *Activate* or *Deactivate*.  
Read Access Logging is now activated or deactivated for the respective app.

## **Related Information**

[How to Define What to Log \[page 2738\]](#)

### 6.3.1.17.2.4.3 How to Compare an Existing Configuration With an SAP Template

You can compare an existing configuration that has been created based on an SAP template with the current SAP template to see if there are any deviations.

#### Context

Because SAP templates are updated from time to time, it can be useful to compare existing configurations that are based on an SAP template. To check for templates with deviations and to see what has changed, follow these steps:

#### Procedure

1. Under *Administration* open *Configuration*.
2. Choose the channel that you want to search through.
3. Under *Search Criteria* select **Deviations from SAP Template** is **True**.
4. Choose *Search*.
5. Choose one of the search results.
6. Choose *More* and select  *Compare with SAP Template*.

##### Note

If you want to rebuild a configuration based on an updated version of the SAP Template, close the pop-up window, choose the configuration, then choose *More* and select  *Rebuild from SAP Template*.

#### Results

A pop-up window opens where you can check which values are different between the SAP Template (left) and the existing configuration (right).

### 6.3.1.17.2.4.4 Channel-Specific Information

The Read Access Logging framework handles the channels generically, but each channel configuration is specific.

The following table shows the objects for each channel that can be logged.

## Channels Available to the Read Access Logging Framework

Channel	Data Logged
Remote Function Call	<p>Remote-enabled function modules:</p> <p>Read Access Logging takes place when data leaves the system.</p> <ul style="list-style-type: none"><li>• Asynchronous outgoing request calls on client side. Logging takes place before the call is made.</li><li>• Synchronous requests on the provider side are logged with their response and fault messages to ensure that the response can be understood. Logging takes place before the call is made.</li></ul>
Dynpro	<p>Configuration is based on Dynpro recordings that list the UI elements from a Dynpro application. Read access to these UI elements is then logged.</p> <p>For more information, see:</p> <ul style="list-style-type: none"><li>• <a href="#">Dynpro Screen Elements that Can Be Logged [page 2748]</a></li><li>• <a href="#">Tips for Configurations with Log Contexts [page 2751]</a></li></ul>
Web Dynpro	<p>Configuration is based on Web Dynpro recordings that contain UI elements from a Web Dynpro application. When recording, UI elements from more than one Web Dynpro application can be gathered, but each configuration is based on UI elements from only one Web Dynpro application within a recording.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p><b>① Note</b></p><p>Web Dynpro applications are technically identified by their Web Dynpro application configurations.</p></div> <p>For more information, see:</p> <ul style="list-style-type: none"><li>• <a href="#">Web Dynpro UI Elements that Can Be Logged [page 2745]</a></li><li>• <a href="#">Tips for Configurations with Log Contexts [page 2751]</a></li></ul>

Channel	Data Logged
Web Service	<p>Web service operations:</p> <ul style="list-style-type: none"> <li>• Synchronous and asynchronous outgoing request messages on consumer side.</li> <li>• Synchronous requests on the provider side are logged with their response and fault messages in a single log entry to ensure that the response can be understood.</li> </ul> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>Synchronous requests on the provider side are logged with their response and fault messages in a single log entry to ensure that the response can be understood.</p> <p>Asynchronous requests on the provider side are logged after the message passes authentication. Waiting until after authentication prevents denial of service (DOS) attacks from being logged. Log entries are created independently of the processing of messages by applications. Log entries are created and remain in the log, irrespective of whether an error is subsequently caused in the application or if the message is later canceled.</p> <p>Local shortcut Web service calls are not logged, since these calls do not leave the system.</p> </div>
ALE/IDoc Communication	<p>IDoc outbound</p> <p>Supported are the following default IDoc outbound protocols:</p> <ul style="list-style-type: none"> <li>• tRFC 3.0/3.1,4.x</li> <li>• qRFC</li> <li>• File, XML file</li> <li>• http, SOAP</li> </ul> <p>Read Access Logging takes place when IDoc data leaves the system, right before the data is handed over to the single communication protocols.</p>
SAP BW	<ul style="list-style-type: none"> <li>• Query requests</li> <li>• Query results or outputs</li> <li>• Value help</li> <li>• Master data maintenance data</li> <li>• Data preview of InfoProviders</li> </ul>
KPro	<p>Read Access Logging takes place when Knowledge Provider (KPro) is used to access the documents.</p> <p>Logging is done for specific PHIO classes for which the Read Access Logging is activated.</p>

Channel	Data Logged
Output Forms (DDIC Interface)	<p>Read Access Logging takes place when an Output Form is rendered for Print Output, Preview or next processing by application. You can configure logging for the name of the Output Form as well as for its fields. Note that the name of the form is logged as the field <code>FORM_NAME</code>.</p>
SAP Gateway (OData)	<p>For OData version 2 (V2) and version 4 (V4) both output logging and input logging is supported: In the RAL configuration users can define the requests for which a service should be logged, so the data that is returned by an OData service. The data which is marked as confidential, is logged.</p> <p>For OData V4 compared to OData V2, some channel fields have been changed/added.</p> <p>For more information, see:</p> <ul style="list-style-type: none"> <li>• <a href="#">Read Access Logging (RAL) and OData V2</a></li> <li>• <a href="#">Read Access Logging (RAL) and OData V4</a></li> </ul>
Electronic Document	<ul style="list-style-type: none"> <li>• Administrative eDocument data</li> <li>• Administrative eDocument file data</li> <li>• eDocument type-specific fields</li> <li>• eDocument file content data: <ul style="list-style-type: none"> <li>• If the file structure is filled, then all the fields from the file structure are available for logging.</li> <li>• If no file structure is defined, then all the data can be logged without its structure (for example, a PDF file). The Electronic Document channel provides three fields for domain assignment.</li> </ul> </li> </ul>

### 6.3.1.17.2.4.4.1 Web Dynpro UI Elements that Can Be Logged

#### Structure

##### ⓘ Note

- Both output and input elements can be logged, but they must be context-bound UI elements.
- UI elements that are generated automatically (for example, when using F4 Help) can be logged. They do not need to have a UI element ID assigned during application development. In these cases, a temporary ID is assigned during the recording.

The table below shows all UI elements that can be logged in Web Dynpro applications. It shows the properties that can be recorded for each UI element.

UI Element	Property
TextEdit	Value
TextView	Text
FormattedTextEdit	Value
FormattedTextView	Text  If <i>value suggest</i> is active, all proposed texts for this UI element that are displayed on the UI are logged.
InputField	Value  If <i>value suggest</i> is active, all proposed texts for this UI element that are displayed on the UI are logged.  Historical values are not recorded.
ToolbarInputField	Value
DropdownByIndex	Texts
Caption	Text
Button	Text
ToolbarButton	Text
ToolbarDropdownByIndex	Texts
LinkChoice	Text
ToolbarLinkChoice	Text
ButtonChoice	Text
ToolbarBtnChoice	Text
ToggleButton	Text
ToolbarToggleButton	Text
Checkbox	Text
Label	Text
LinkToAction	Text
LinkToURL	Text

UI Element	Property
ToolbarLinkToAction	Text
ToolbarLinkToURL	Text
FileDownload	fileName, mimeType, text, data
Panel	Title
RadioButton	Text Selected Key
RadioButtonGroupByIndex	Text
RadioButtonGroupByKey	Selected Key
ToggleLink	Text
TriStateCheckbox	Text
SectionHeader	Text
DropdownByKey	Selected Key
ToolbarDropdownByKey	Selected Key
DropdownListBox	ItemText ItemDescription Texts
ToolbarDropdownListBox	ItemText ItemDescription Texts

The constituents of the following multiple UI elements can be logged with Read Access Logging:

- Table
- CTable
- RowRepeater
- MultiPane
- Accordion and MultiAccordionItem

## Related Information

[Channel-Specific Information \[page 2742\]](#)

[Channel-Specific Information \[page 2742\]](#)

## 6.3.1.17.2.4.4.2 Dynpro Screen Elements that Can Be Logged

A list of screen elements in Dynpro applications that can have read access logged.

The main scope of Read Access Logging is to log output fields on result screens. Optionally, also the input fields of the corresponding selection or query screens can be logged. Logging and defining a log context are supported for such selection or query screens and their result screens if both are implemented and processed in the same program. Please note that one business transaction may be processed using more than one program. In such cases, logging of the input is not possible.

The table below shows the basic screen elements that can be logged in Dynpro applications.

Basic Screen Element	Comment
Input fields	Value of input field
Output fields	Value of output field
Radio buttons	For each radio button, a value is logged (true/false). Note that each radio button is logged separately. There is no logging of entire groups of radio buttons.
Checkboxes	Value of checkbox
Dropdown lists	Technical key value with marker for selected key. Note that the texts from the user interface are not logged.
Table controls	Columns of table control
Parameters	Value of the parameters of a selection screen
Subscreens	As subscreens consist of the same elements as screens, logging takes place for the elements of subscreens in the same way as for screens

Basic Screen Element	Comment
Select-Options	<p>Value of each select-option of a selection screen (list of sign, option, low/high values).</p> <p>A select-option <code>SELOPT</code> is displayed as:</p> <ul style="list-style-type: none"> <li>• a <code>LOW</code> or <code>HIGH</code> field for a single selection</li> <li>• a push-button (arrow icon) for a multiple selection that is represented as a table of fields (<code>SIGN</code>, <code>OPTION</code>, <code>LOW</code>, <code>HIGH</code>)</li> </ul> <p>To be able to log <code>SELOPT-LOW</code> or <code>SELOPT-HIGH</code> fields, they have to be configured.</p> <p>You can enter the data to <code>SELOPT-LOW</code> or <code>SELOPT-HIGH</code> manually:</p> <ul style="list-style-type: none"> <li>• If a program was executed without a screen round trip, the fields <code>SELOPT-LOW</code> and <code>SELOPT-HIGH</code> can be logged as input fields.</li> <li>• The multiple selection is attached to the <code>SELOPT-LOW</code> field: if the field <code>SELOPT-LOW</code> is configured for logging, the multiple selection is configured for logging as well. If a screen round trip was done (e.g. pressing <code>Enter</code>), or if the multiple selection was used, the <code>SELOPT</code> data is added to the multiple selection table, therefore the <code>SELOPT-LOW</code> field can be logged as an input or output field.</li> </ul> <p>If a variant that is used to provide the input data for a selection-option shall be logged, it has to be configured for output logging. Input data provided for the multiple selection can be logged, even if it was not displayed.</p> <p>Input data provided via variant for other parameter fields on a selection screen can be logged using the input logging.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p><b>① Note</b></p> <p>To be sure that all the data related to the select-option field is logged, it should be configured for both input and output logging.</p> </div>
Value help (F4)	Entries from the value help (F4 help) can be recorded by choosing the read access logging function from the context menu.

The table below shows the screen attributes that can be logged:

Screen Attribute	Comment
Screen title	The title of a screen

Screen Attribute	Comment
Transaction code	The value of a transaction code at runtime
OK code	Code that is set when triggering an action (for example, clicking a button)
Message	The status message displayed on a screen

The table below shows the controls that can be logged:

Control	Comment
ALV grid	Columns of a display-only ALV grid control
Tab strip	Same logging behavior as for subscreens. If switching tabs happens on front-end, logging is done together for all tabs. If, on the other hand, switching tabs happens on application level, the tabs are logged separately. In addition, an OK code can only be logged when switching tabs happens on application level.

The table below shows the additional screen elements that can be logged:

Screen Element	Comment
ABAP Lists	ABAP lists can be recorded by choosing the read access logging function from the context menu. The pseudo field <code>\$_LIST_CODE</code> will be added to the created recording. Using this pseudo field, all lists of a program can be configured for logging. It is not possible to log input fields and OK codes of a list.

It is not possible to log other screen objects, such as the following:

- Input-enabled ALV grid
- ALV tree
- File download

For more information about the availability of features of Read Access Logging, see SAP Note [1969086](#).

## Related Information

[Channel-Specific Information \[page 2742\]](#)

[Channel-Specific Information \[page 2742\]](#)

[How to Manage Recordings of Application User Interfaces \[page 2736\]](#)

## 6.3.1.17.2.4.4.3 Tips for Configurations with Log Contexts

### Definition

In a Read Access Logging configuration for Dynpro and Web Dynpro channels, you might need to use a log context to make sure that your log can be interpreted correctly. The log context is the UI element that other UI elements within the logging session depend on. When read access is being logged and the field that is defined as the log context changes on the user interface, all other fields are deleted from memory before the next log entry is created. Whether or not you need a log context depends on the user interface of your application. The graphic below shows a screen sequence taken from a simple HR application and the accompanying table gives recommendations on how to configure Read Access Logging for this screen sequence with the help of the log context.

#### Note

Log contexts are based on single applications or programs, like configurations. Log contexts cannot be applied across several configurations or applications/programs.

For Dynpro, single transactions can contain more than one program, which may not be apparent to you when you create a recording. In this case too, log contexts cannot be spread over multiple programs used in the transaction.

### Example

The graphic below shows a sequence of three screens. On Screen 1, the user enters an employee number. Screen 2 displays the social security number and religion of the employee. Screen 3 displays the salary of the employee.



The table below shows how to configure the screen sequence depicted in the graphic above. Note that the field **Employee Number** is a different field on the three screens; on the first screen, it is an input field and on the second and third field, it is an output field like the other fields that cannot be edited.

Screen Number	UI Element	Recommended Logging	
		Type	Comment
Screen 1	Employee No	Log with value and define as log context	<p>The employee number is an input field, we recommend that you log it. As the other fields on Screen 2 depend on the employee number, we recommend defining it as the log context.</p> <p>In the log, the field that is the log context is marked for easy identification.</p>
Screen 2	Employee No	No logging necessary	<p>The employee number has been entered on Screen 1 and is repeated here as an output field. As you have already specified it as the log context, you do not need to log this field.</p>
		<p><b>① Note</b></p> <p>Even if the employee number is not repeated on each screen, the log context is still logged as part of the log entry of these screens.</p>	
Social Security No		Log without value (for example, legal regulations may prohibit logging social security numbers)	<p>The social security number of this employee is displayed within the same log entry as the log context. If you configure to log it without the value, only the fact that the field was accessed is logged, but not the value.</p>
Religion		Log with value	<p>The religion of this employee is displayed within the same log entry. If you configure to log it with value, the religion is displayed in the log.</p>

Screen Number	UI Element	Recommended Logging	
		Type	Comment
Screen 3	Employee No	No logging necessary	As you have specified the employee number from Screen 1 as the log context, you do not need to log this field.
		<p><b>① Note</b></p> <p>Even if the employee number is not repeated on each screen, the log context is still logged as part of the log entry of these screens.</p>	
	Salary	Log with value	As it is the only field that is configured on this screen, the salary of this employee will be displayed with the log context from Screen 1 in one log entry.

## Result

When a user clicks through this screen sequence when it is configured as described above, two log entries are created, one for Screen 2 and one for Screen 3. On Screen 1, no output field exists that can be logged. On Screen 3, the employee number does not have to be logged as the employee number field from screen 2 is the log context and is thus automatically logged during a screen sequence even if it is not displayed on subsequent screens.

## Related Information

[Channel-Specific Information \[page 2742\]](#)

[How to Define What to Log \[page 2738\]](#)

## 6.3.1.17.2.5 How to Exclude Specific Users from Read Access Logging

### Context

You cannot exclude specific users from Read Access Logging by using the app Read Access Logging Configuration.

### Procedure

- To let users be excluded from Read Access Logging, please contact the Service Center.

## 6.3.1.17.2.6 How to Check Runtime Errors and See Changes to Configurations

### Context

The Administrative Log of Read Access Logging provides information about different types of information such as:

- Action - changes made to Read Access Logging configuration.
- Error - errors that occur within the Read Access Logging framework.

### Action Log Entries

The action log records the following Read Access Logging changes:

- Configuration
- User exclusion list
- Enabling and disabling of Read Access Logging in the current client

#### Note

Only configuration changes performed manually are recorded in the log. Configurations that have been transported into the system are not included in the Administrative Log.

## Error Log Entries

Error entries could be errors in the configuration, or problems that occur during logging. The log entry may recommend action to resolve the error and/or provide other information relevant for key users or SAP Active Global Support.

## Procedure

1. In the app Read Access Logging Configuration, open [Administrative Log](#).
2. Select a *Data Source*, either the database or the archive.
3. For *Databases*, specify a time filter. For *Archives*, select the *Archive Name*.
4. Enter a *User Name* to search the log for configuration actions or read access performed by a specific user.
5. Specify a log *Type* for archives or *Event Type* for databases (*Error*, *Warning*, *Message*, *Action* or *Logging Error*).

## 6.3.1.17.2.7 Transporting Read Access Logging Data

RAL objects that you've created can be transported from a Q system to a P system (development system to production system) with the *Export Customizing Transports* app

In your RAL application, you can create different customizing objects like configurations or recordings. These objects are added to the default customizing request that you can export from a Q system and import into a P system. The objects that come from the RAL application are:

- Configurations  
Refer to SAP note [2963664](#) for details.
- Log domains
- Purposes
- Recordings

To transport your changes from the development system to the productive system, perform the following steps:

### In your development system:

1. Use the *Manage Software Components* app to make sure there is a cloned business configuration software component. See [Manage Software Components](#).  
In case it is not available yet, you need to create a business configuration software component and clone it.  
See [How to Create Software Components](#) and [How to Clone Software Components](#).
2. Make sure that there is an open customizing transport request for your system and that the customizing transport request belongs to the cloned business configuration software component.  
To create a customizing transport request, you use the *Export Customizing Transport* app. You need the *SAP\_BR\_BPC\_EXPERT* role to access this app. See [Export Customizing Transports \[page 2582\]](#).  
If a software component of the type Business Configuration is available in the system, the created request will target this software component. If no such software component is available, the request will be created without any target.

3. Make the actual changes to the RAL objects.
4. Release the task and customizing transport request in the *Export Customizing Transport* app.

#### In your productive system:

1. In the Manage Software Components app, pull the changes for the business configuration. See [How to Pull Software Components](#).
2. Check in the *Read Access Logging Configuration* app if the changes are visible.

### Related Information

[Working in the Export Customizing Transports App \[page 2584\]](#)

## 6.3.1.17.3 Read Access Logging: Monitor

With this app you can use the read access log for evaluation purposes. In it, all log entries as well as all errors that occurred in Read Access Logging are displayed.

### Key Features

You can use this app to:

- Choose a data source
- Search for log entries
- Evaluate log entries

### Prerequisites

To access the app, you need to have the `SAP_CORE_BC_RAL` business catalog assigned to your user. This business catalog is contained in the `SAP_BR_DATA_PRIVACY_SPECIALIST` business role template.

### Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SEC-RAL.

### 6.3.1.17.3.1 How to Monitor the Read Access Log

#### Context

You use the read access log for evaluation purposes. In it, all log entries as well as all errors that occurred in Read Access Logging are displayed. There are the following data sources from which you can display log entries:

- *Expanded Database* (default)
- *Expanded Archive*
- *Raw Database*  
Only contains the Read Access Logging data of the current client.
- *Raw Archive*  
Archive of the raw database.
- *Raw Archive with Index*  
Indexed archive of the raw database.

#### Procedure

1. Open the Read Access Logging: Monitor app.
2. Select the data source from which you want to display log entries.
3. Search for the log entries you want to evaluate.

To search for values of fields that have been accessed, you must use the *Expanded Database* as the source.

For data source *Archive*, select the archive in which you want to search for log entries. Additionally, there are various search criteria that you can use (choose the *Add Line* icon). Typically, you would search for a time range and a log domain or a logging purpose. These search criteria are also available if you search using the database as the source.

#### ⓘ Note

Searching the archive may be time-consuming. It is much faster if you search an archive with an index. In this case, the search criteria that you have specified for the index are available.

When you choose *Search*, all log entries are displayed that match your search criteria. By default, only a few attributes are displayed per log entry. You can switch between the default and the extended view.

Within the search, you can clear the values you have entered for search criteria by choosing *Clear*, and you can reset the search criteria to the default by choosing *Reset*.

**ⓘ Note**

You can use the button *Download* at the top left of the search result table to save your search results in an Excel file.

4. Select a log entry to display detailed information at the bottom.

On the *Details* tab, all fields are displayed with their log domain and, if logged, the field values. On the *Access Environment* tab, you find more information about the underlying object that was logged.

### 6.3.1.17.4 Integrating Read Access Logs

Service name: SRAL\_API\_RAW\_LOG

This service enables you to retrieve the raw Read Access Logs. You can use the log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

#### Technical Details

Service Group (incl. Name-space if Existent)	Repository ID	Service Name (incl. Name-space if Existent)	Version
SRAL_API_RAW_LOG	SRVD_A2X	SRAL_API_RAW_LOG	0001

#### Service Structure

##### Service Header (optional)

The service header contains information about the service.

##### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
ReadAccessLogRAW	Read Access Log in RAW format		<a href="#">Retrieving Security Audit Log [page 2777]</a>

## Service Response

The details included in the response vary according to the type of operation. For more information, see Operation for Read Access Log Integration.

## Integration with SIEM Solution

Service name: SRAL\_API\_RAW\_LOG

Communication scenario: SAP\_COM\_0915

This service enables you to retrieve the Read Access Log data. This service enables you to retrieve the raw Read Access Logs. You can use the log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

### 6.3.1.18 Security

[Secure Communication \[page 2759\]](#)

[Display Security Audit Log \[page 2775\]](#)

#### 6.3.1.18.1 Secure Communication

SAP BTP, ABAP environment offers secure communication features for securing both machine-to-machine communication (server communication security) as well as communication between systems and end-user UIs (frontend communication security).

Communication security is maintained on multiple levels.

#### Establishing the Communication Channel to a Host

The connection is initiated by a client calling a host to set up a cryptographically secured communication channel. This is known as a TLS (or SSL) handshake. During the handshake, the host proves its identity by presenting a server certificate. The client validates the server certificate to gain assurance that the connected host coincides with the host the client intended to call. This prevents man-in-the-middle attacks.

In case of a web UI, the validation of server certificates is done in the browser using a list of trusted root certificates provided by the browser or the operating system.

The certificate of your system is signed by DigiCert Global Root CA.

In other cases, the client may also be a backend server talking to another backend server. Here, the server initiating the connection takes the role of the client and the other one is the host.

## Authentication of the Client/User

After the communication channel is established, the application on the host side may verify the caller's identity if the application requires authentication. The most common method for this is still basic authentication (user/password). A more secure option is certificate-based authentication (mTLS). Here, the client presents an X.509 certificate to the application to prove its identity.

## Additional Protection of Web Clients

For web clients, there are additional mechanisms protecting the user from attacks like cross site scripting (XSS) or redirect attacks. These mechanisms are defined by the server, but enforced in the user's client.

### 6.3.1.18.1.1 Frontend Communication Security

Communications between the customer browser and the system landscapes of SAP BTP, ABAP environment are secured by industry best practices and state-of-the-art open cryptographic standards. Customers use a unique, customer-specific URL. Communication is carried out via the Reverse Proxy (RP) component. The communication channels are secured by using Transport Layer Security (TLS) protocol version 1.2.

Client-side security controls implemented by the browser and by SAP BTP, ABAP environment mitigate the risk of various attacks, including cross-site scripting, data injection, and clickjacking.

Some of these controls can be configured with allowlists such as the following:

- Trusted sources of scripts, stylesheets, and fonts can be added to the default Content Security Policy.
- Trusted hosts can be defined for the clickjacking protection, and for safe redirects.

#### 6.3.1.18.1.1.1 Content Security Policy

Content Security Policy (<https://www.w3.org/TR/CSP3>) is a standard which allows you to disable certain HTML/JavaScript features to reduce the attack surface of applications running in a browser (for example, as a second line of defense against cross-site scripting attacks). The policy explicitly lists all allowed sources from where resources (scripts, styles and fonts) can be loaded.

Servers can define different content security policies (CSPs) for different paths (applications) and different resource types to provide very granular control.

CSPs can be defined in reporting or blocking mode. In reporting mode, a violation will be reported to a URI specified within the CSP. In blocking mode, violations are not only reported but execution of violating code or loading of the violating resource is additionally blocked by the browser.

## Related Information

[Manage Content Security Policy \[page 2765\]](#)

### 6.3.1.18.1.1.2 Clickjacking Protection

Clickjacking (or UI redressing) is a type of attack that tricks the user into triggering actions within an application by hijacking mouse clicks or other user inputs.

In the simplest attack scenario, an invisible iFrame containing the attacked page is positioned over an attacker-controlled page. The user thinks the UI is original and is unaware of the actual actions triggered on the invisible page in the frame. To protect against this attack, you need to control whether to render your application within a frame and which pages are allowed.

The allowlist service is an ABAP-wide service and supports HTML-based frameworks to implement protections. By default, clickjacking protection is active.

As soon as the protection is enabled, a special check is performed every time an application is rendered. If the application is embedded into another one, the check determines whether the other application is secure. If the check fails, the embedded application is not framed. An error message appears.

An application is considered secure if one of the following applies:

- The application itself is not embedded in another frame
- The host of the application is part of the same domain as the embedding applications (same origin policy).
- The host of the application is part of the allowlist. For multi-domain scenarios, you have to make sure that the host name of the application in which your application is embedded, is part of the allowlist.

## Related Information

[Maintain Protection Allowlists \[page 2763\]](#)

## 6.3.1.18.1.1.3 Trusted Network Zones

### Redirection Attacks

The primary use of trusted network zones is to prevent redirection attacks.

In a redirection attack, the attacker tricks the victim into sending a specially crafted request to the server. The server then redirects the request to a malicious website based on the information in the request.

This is used for phishing. The victim might not be aware of the redirect because the initial request was sent to a trustworthy system and the malicious site might be built to resemble the trustworthy system.

The configuration of trusted network zones prevents such attacks by limiting which URLs the server can redirect to.

### Upload Limitation

The trusted network zones configuration also restricts uploads defined by URL.

Where file upload is possible by specifying a URL rather than a file on the filesystem, the entered URL is checked against the trusted network zones list. Note that host, port, and scheme of the URL must be maintained in [Trusted Network Zones](#) for the upload to be permitted.

### Related Information

[Maintain Protection Allowlists \[page 2763\]](#)

## 6.3.1.18.1.1.4 Session Timeout

Both web applications in the SAP Fiori Launchpad and in SAP Cloud Identity Services - Identity Authentication log out users after a certain period of inactivity. The length of this period can be changed independently for the two.

### SAP Fiori Launchpad

Users are automatically signed out from the SAP Fiori Launchpad after a period of inactivity that is configurable in the [Manage Launchpad Settings](#) app.

### Note

The following timeout parameters are visible in the [Manage Launchpad Settings](#) app. If you need to make changes, we suggest you keep the value of the SESSION\_TIMEOUT\_INTERVAL\_IN\_MINUTES parameter synchronized with the timeout in your Identity Authentication tenant so as to avoid inconsistencies in session management.

- SESSION\_TIMEOUT\_INTERVAL\_IN\_MINUTES
- SESSION\_TIMEOUT\_REMINDER\_IN\_MINUTES
- SESSION\_TIMEOUT\_STOP\_DATA\_REFRESH

## Related Information

[Manage Launchpad Settings](#)

[Signing In and Signing Out](#)

[Configuring the Automatic Sign-Out](#)

[Configure Session Timeout \(SAP Cloud Identity Services - Identity Authentication\)](#)

## 6.3.1.18.1.1.5 Apps for Configuring Frontend Communication Security

The following apps are used for configuring frontend communication security:

- [Maintain Protection Allowlists \[page 2763\]](#)
- [Manage Content Security Policy \[page 2765\]](#)

### 6.3.1.18.1.1.5.1 Maintain Protection Allowlists

With this app you can maintain a list of trusted hosts or URL patterns.

With this app you can define secure applications by adding trusted hosts to allowlists. The following allowlists are available:

- [\*Clickjacking Protection\*](#)

By default, clickjacking protection is active to protect your systems against malicious clickjacking. If the system is embedded into another application, the check determines whether the other application is secure. To add trusted hosts, you have to enter specific details, such as schema and port, for each trusted host to make sure that malicious hosts are identified and prevented from calling your system.

- [\*Trusted Network Zones\*](#)

Here you can list URL patterns for which a redirect is allowed or blocked.

- [\*Trusted CSS Style Sheets\*](#)

Here you can list URL patterns for loading CSS style sheets. This may be necessary if customer extensions are using external CSS style sheets.

## Key Features

You can use this app to:

- Add trusted hosts to the required allowlist
- Edit trusted hosts or URL patterns
- Delete trusted hosts or URL patterns

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.18.1.1.5.1.1 How to Add Trusted Hosts for Cross-Origin Resource Sharing (CORS)

#### Context

CORS is a mechanism that allows web browsers or other web clients access to your sites. Access of this kind is usually forbidden by the Same-Origin-Policy (SOP). To add trusted hosts using CORS, proceed as follows:

#### Procedure

1. Open the *Maintain Protection Allowlists* app on the SAP Fiori Launchpad.
2. Open the *Cross-Origin Resource Sharing* tab.
3. Enter the *Trusted Host Name* (SAP Analytics Cloud, for example, could have the following pattern: *mytenant.us1.sapbusinessobjects.cloud*).
4. Enter the *HTTP Service Path* (for example */sap/opu/odata/sap/APS\_IAM\_API\_BROLE\_CDOC*).
5. Select the allowed HTTP methods .
6. Enter the allowed headers (for example **content-type**)

Only the following response headers are generally accessible in a CORS query:

- *Cache Control*
- *Content Type*
- *Last Modified*
- *Content Language*
- *Expires*
- *Pragma*

If the server wants to give the client access to further headers, it has to use the *Exposed Headers* option.

7. In addition, you can define exposed headers under *Optional Response Headers* if required. Please select the required checkboxes.
8. Click *Add*.

## Related Information

[Understanding the Same-Origin Policy and CORS](#)

### 6.3.1.18.1.1.5.2 Manage Content Security Policy

Content Security Policy is a standard that allows to disable certain HTML/Javascript features to reduce the attack surface of applications running in a browser (for example as second line of defense against cross-site scripting attacks).

With this app you can view an allowlist of trusted sources. You can add your own trusted content for example if you have developed your own SAPUI5 app that loads external resources such as fonts, scripts or styles. Moreover, you can display any violations of the policy in a log.

## Key Features

You can use this app to:

- Display an allowlist of trusted sources
- Add new trusted sources to the allowlist  
For fonts, use UI\_RESOURCES\_FONTS; for scripts, use UI\_RESOURCES\_SCRIPTS, for styles, use UI\_RESOURCES\_STYLES.
- Display logs listing violations  
Please note the following information about browser-related effects that in some cases might result in additional violations while in other cases violations are not listed:
  - The violation logs might contain records indicating that an EVAL violation has occurred. These violations can be ignored. They are caused by unexpected behavior in Google Chrome.
  - Firefox does not send the session cookie with the reporting requests. These requests will therefore be rejected by the backend, and are thus not included in the log.

- Browser extensions sometimes insert non-policy conform code into an HTML page. This results in violation log entries that are not caused by the applications themselves.

#### ⓘ Note

You can switch the reporting endpoint delivered by SAP on and off. To do this, activate or deactivate the *Reporting Endpoint Status* switch on the *Settings* tab.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.18.1.2 Server Communication Security

In integration scenarios, the SAP BTP, ABAP environment backend exchanges data with other systems. These systems may be SAP or third-party systems in the cloud or on-premise.

To secure these system-system communication channels, authentication based on certificates can be used.

We distinguish between outbound and inbound integrations.

## Outbound Integrations

In outbound integrations, the SAP BTP, ABAP environment calls an external system. To establish the connection, the external system must present a trusted (server) certificate.

The SAP BTP, ABAP environment may in turn authenticate towards the external system by presenting a client certificate that the external system trusts. Depending on the capabilities and configuration of the external system, other authentication methods like OAuth or user/password are possible.

## Inbound Integrations

In inbound integrations, an external system calls the SAP BTP, ABAP environment backend system. The authentication method is specified as part of the communication arrangement. In case of certificate-based

authentication, the external system must present a specific certificate that is maintained in the communication arrangement and signed by an approved CA.

## 6.3.1.18.1.2.1 Secure Communication for Outbound Integration

Secure communication is required in all integration scenarios that connect SAP BTP, ABAP environment to other systems.

These outbound integration scenarios may include:

- other SAP cloud systems
- your on-premise systems
- third party systems (cloud, non-cloud)

When a connection between SAP BTP, ABAP environment and the external system is established, this happens in two steps. In the first step, the SAP BTP, ABAP environment validates the identity of the external system to ensure there's no man in the middle impersonating the external system. In the second step, SAP BTP, ABAP environment authenticates against the external system (if required by the external system).

### Step 1: SAP BTP, ABAP environment Verifies the Identity of the External System

When establishing the secure communication, the external system must prove its identity using a server certificate that is signed by a trusted certificate authority (CA).

For secure communication to SAP-owned systems and services, SAP BTP, ABAP environment contains a preconfigured list of trusted CAs (marked as *Managed By SAP*, not changeable by customers).

For integration to non-SAP systems, you can maintain the list of trusted CAs (*Managed By Customer*) in the [Maintain Certificate Trust List](#) app.

#### Expiration of Certificates of Trusted CAs

When a certificate from the list of trusted certificates reaches its expiration date, it needs to be deleted from the list in your SAP BTP, ABAP environment. You can do this using the [Maintain Certificate Trust List](#) app.

In case of certificates that are managed by SAP, you need to update the list of certificates in the app. You can choose between manual and automatic updating. When you check for updates manually, you will be prompted to select which updates you'd like to apply, if any are available. If you choose to set up automatic updating instead, you can select if you'd like to apply updates regarding both newly introduced and deleted certificates, or only new certificates. These changes will be applied by the system each time the global certificate trust list is updated by SAP.

Certificates that have been added manually need to be deleted manually as well.

### → Remember

If your external system that SAP BTP, ABAP environment communicates with uses a certificate that is signed with an expiring root CA certificate, remember to change that certificate. Otherwise, the communication will fail.

## Step 2: SAP BTP, ABAP environment Authenticates as Client

Different methods can be used to authenticate. The availability of these options depends on the capabilities of the external system.

The credentials for outbound communication are configured in the [Maintain Communication Systems](#) app.

### Authentication Using a Client Certificate (mTLS)

For authentication against an external system, SAP BTP, ABAP environment may use the certificate-based authentication method. In this method, SAP BTP, ABAP environment presents a client certificate to the external system.

The client certificate is linked to a private key that serves as a secret for identifying SAP BTP, ABAP environment. The client certificates must be trusted by the target system or be signed by a CA the target system trusts.

#### *Expiration of Client Certificates and the Default Client Certificate*

When client certificates expire, the external system will not accept them for authentication anymore. Consequently, your SAP BTP, ABAP environment will not be able to connect to the external system anymore. You must update the trust between the system in time before this happens by uploading a new client certificate (private key) in the [Maintain Client Certificates](#) app. The corresponding public key must be made known to the external system.

### Authentication Using OAuth

OAuth-based authentication is a two-step process (the so-called “client credential flow”). In the first step, the SAP BTP, ABAP environment contacts a token provider to obtain an access token. During this step, the SAP BTP, ABAP environment authenticates against the token provider using basic authentication using client ID and client secret or mTLS using a client certificate (only available in OAuth 2.0). The authentication endpoints of the token provider are maintained in the [Communication Systems](#) app.

As the second step, the call to the external system is made and the access token is passed to the external system.

### Authentication Using User Name and Password

SAP BTP, ABAP environment also supports basic authentication (user name and password). As there is a high risk of the credentials leaking, we recommend using certificate-based authentication instead.

### No Authentication

Some external systems do not require authentication. They just offer public endpoints available to everybody.

## Related Information

[Maintain Certificate Trust List \[page 2771\]](#)

*Maintain Client Certificates*

[Communication Systems \[page 2598\]](#)

### 6.3.1.18.1.2.2 Secure Communication for Inbound Integration

Integration scenarios from a customer system to SAP BTP, ABAP environment (inbound integration) require secure communication.

You specify and configure the authentication method in the *Communication Arrangements* app when you select a user for inbound communication. The underlying communication scenario defines the authentication methods that are available for a communication arrangement.

Communication scenarios can offer the following types of authentication method:

- Token-based
- Certificate-based
- Basic (User ID and Password)

Token-based authentication methods enable principal propagation, which means that the resulting sessions run in the context of a business user. In contrast, sessions using certificate-based or basic authentication run in the context of a technical communication user.

#### Token-Based Authentication

Token-based authentication methods are configured in the *Communication System* app, where you establish trust with the token provider and define which part of the token contains the user name for principal propagation.

##### Note

In system-to-system communication of the kind covered here, SAML and OIDC authentication flows use the bearer flow mechanism. These authentication flows differ from flows implemented in browser-based SAML 2.0 and OIDC authentication because there's no UI or user interaction in system-to-system communication.

#### OAuth-Based Authentication

When the external system authenticates to SAP BTP, ABAP environment using OAuth, it presents a short-lived token to SAP BTP, ABAP environment. This token is cryptographically signed by a trusted token provider.

To establish the trust relationship between SAP BTP, ABAP environment and the token provider, you must upload the token provider's certificate (public key) to your SAP BTP, ABAP environment in the *Communication Systems* app.

## SAML-Based Authentication

In SAML-based authentication, the external system authenticates by passing a SAML assertion (a cryptographically signed piece of text) to SAP BTP, ABAP environment.

For authentication to succeed, the issuer of the SAML assertion must be trusted by the SAP BTP, ABAP environment. To ensure trust, you must upload the certificate (public key) of the SAML issuer in the [Communication Systems](#) app. This enables SAP BTP, ABAP environment to verify the authenticity of the SAML assertion.

## OpenID Connect (OIDC) Authentication

OpenID Connect standardizes the way in which user information is transferred via JSON Web Token (JWT) during login. To ensure trust with the token provider, you must upload a document containing the provider's metadata. This document can include the URL for fetching the OIDC provider's private keys (Web Keys), allowing trust to be established automatically. Alternatively, you can configure the URL for downloading the OIDC provider's public key manually or enter the keys directly.

## Certificate-Based Authentication

For certificate-based authentication, the client certificate (public key) of the external system must be uploaded to the communication user used in the communication arrangement.

Such client certificates must be signed by approved certification authorities (CAs).

If the certificate is signed by an untrusted CA, establishing a connection fails during the TLS handshake.

If the external system presents a certificate that is signed by a trusted CA, but doesn't match the certificate configured in the communication arrangement, authentication fails at the application level.

A list of all root CAs approved by SAP Global Security is available in SAP Note [2801396](#) (SAP Global Trust List).

## Certificate Expiration

Every certificate is valid for a certain period. Once a certificate expires, it can't be used for authentication anymore.

When a certificate of a root CA expires and is replaced by a new certificate, you also need to consider any certificates that were signed by the expiring CA certificate. These certificates have to be replaced by ones that signed with the new CA certificate. Remember to update your communication users accordingly.

## Basic Authentication (User ID and Password)

To use password-based authentication, set a secure password for the communication user in the [Maintain Communication Users](#) app.

### Caution

SAP highly recommends using strong authentication methods such as token- or certificate-based authentication for greater security.

## Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

[SAP Note 2801396](#)

### 6.3.1.18.1.2.3 Apps for Configuring Server Communication Security

The following apps are used for configuring server communication security:

- [Maintain Certificate Trust List \[page 2771\]](#)
- [Maintain Client Certificates \[page 2772\]](#)

#### 6.3.1.18.1.2.3.1 Maintain Certificate Trust List

With this app you can maintain a list of trusted certificates. If certificates of communication partners are classified as trusted, outbound communication to these partners can be enabled.

With this app you can monitor all available trusted certificates.

#### Key Features

You can use this app to:

- Display a list of all already existing trusted certificates
- Upload a new certificate
- Display detailed information
- Check for updates of certificate trusts and adopt the changes if required.
- Automatically apply all changes to the SAP trust list in your trust list after an upgrade. You can either just add new certificates or add new certificates and simultaneously remove deleted certificates.
- Delete trusted certificates from the list. This feature is enabled only for the certificate type *Managed By Customer*. Certificates of the type *Managed By SAP* cannot be deleted and therefore the *Delete* button is disabled.

#### Supported Device Types

- Desktop

- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

### 6.3.1.18.1.2.3.2 Maintain Client Certificates

With this app you can upload client certificates for your area to enable secure outbound certificate-based authentication. You can maintain one central list of certificates you can use when defining communication systems for outbound communication.

#### Key Features

You can use this app to:

- Upload certificates in PKCS #12 format
- Download certificates
- Display general information and usage in communication arrangements and systems
- Replace client certificates
- Create signing requests

#### Note:

Once you have uploaded your certificate in the *Maintain Client Certificates* app, you have to edit it to enable it for usage in SAP BTP. You do this by selecting *Client Default for ABAP on BTP* in the *Certificate Category* dropdown.

#### Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-COM.

## Related Information

[Communication Arrangements \[page 2595\]](#)

### 6.3.1.18.1.2.3.2.1 Using Client Certificate Authentication with SAP BTP Destination Service

To enable client certificate-based authentication in SAP BTP, ABAP environment, the following prerequisites have to be met:

SAP BTP cockpit:

#### For Web Socket RFC

1. Select your Web Socket destination within your used destination service.
2. Under *Additional Properties*, apply the *jco.client.tls\_client\_certificate\_logon property* and set it to value **1**.
3. Choose *Save*.

#### For HTTP

1. Select your HTTP destination within your used destination service.
2. In the *Authentication* dropdown list of the *Destination Configuration* area, select *Client Certificate Authentication*.
3. Choose *Save*.

### 6.3.1.18.1.2.3.2.2 How to Handle Default Client Certificate Renewal

If you use the default client certificate for certificate-based authentication in outbound integration scenarios, you must update the trust between the systems when a new default client certificate is issued.

## Context

In outbound integration scenarios that use certificate-based authentication, the SAP BTP, ABAP environment system needs to authenticate itself against the external system with a client certificate. You can use the default client certificate provided by SAP BTP, ABAP environment for this purpose. This certificate will be rotated twice a year by SAP, regardless of its remaining validity.

The new certificate needs to be configured in your external system that communicates with SAP BTP, ABAP environment.

## Rotation Process

The rotations occur every year on March 1 and September 1, and the process is divided into two stages.

### One Month Before Rotation

On the first day of the month preceding the rotation (February 1 and August 1), a new default client certificate will be issued.

The new default client certificate will be available for export in the **Maintain Client Certificates** app and will be labeled **Client Default**. The expiring certificate will be renamed to **Client Default Expiring**.

Once the new default client certificate is available, follow these steps:

1. **Upload the New Certificate to the External System:** If the external system supports multiple certificates simultaneously, keep the old one until the replacement process is complete:
  - Open the **Maintain Client Certificates** app.
  - Export the new **Client Default** certificate in the format required by the external system.
  - Upload the certificate to the external system.
2. **Rotate the Used Certificate in the Communication System:**
  - In the **Maintain Client Certificates** app you can see by which Communication Systems the **Client Default Expiring** certificate is used.
  - Open the **Communication Systems** app.
  - For each communication system using the **Client Default Expiring** certificate:
    - Navigate to the **Users for Outbound Communication** section.
    - Select the user associated with the **Client Default Expiring** certificate.
    - In the dialog, change the **Client Certificate** value to **Client Default**.

#### ① Note

Switch to the new certificate as soon as possible. This ensures that if any issues arise, there will be time to troubleshoot within the 30-day overlap of the certificate validity periods.

### On the Rotation Date

Once the old certificate is rotated on either March 1 or September 1, it will be removed from the list in the **Maintain Client Certificates** app and all remaining Communication Systems using the old certificate will be migrated to the new Client Default certificate.

### What Happens If You Don't Act

If you do not update your communication users and the new certificate in your external system, the outbound integration scenarios which use the default client certificate for authentication might break. You will get

either the 401 Unauthorized or the 403 Forbidden HTTP status code message when trying to connect, depending on the implementation of the external system.

### 6.3.1.18.2 Display Security Audit Log

With this app, you can display information about security-relevant events that occur in your SAP system. This can be necessary in case of an audit.

#### Key Features

This app provides the following key features:

- Recording of security-relevant events in your SAP system.
- Access to previously specified log files in the form of an audit analysis report.

#### Scope

The Security Audit Log records the following information:

- Security-related changes to the ABAP Platform (for example, change of system changeability)
- Information that provides a higher level of transparency (for example, successful and unsuccessful logon attempts)
- Information that enables the reconstruction of a series of events (for example, successful or unsuccessful transaction starts)

The app *Display Static System Audit* (available for the external auditor role) displays a detailed list of the Security Audit Log events, which are visible in the Security Audit Log app. SAP Note [2903873](#) provides a list of currently recorded Security Audit Log events.

#### Supported Device Types

- Desktop
- Tablet

#### Component for Customer Incidents

If you need support or experience issues, report an incident under component BC-SEC-SAL.

## Related Information

### 6.3.1.18.2.1 Integrating Security Audit Logs

Service name: RSAU\_LOG\_API

This service enables you to retrieve the security audit log data. You can use the audit log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

## Technical Details

A service group contains services belonging to the same business object model and so it shares similar environment conditions. This means the configuration and administration of a service group apply to all services in a service group, that is, routing, authorization, and so on, so you only have to do them once.

In the OData version 4 (V4) runtime implementation of the SAP Gateway Foundation, framework services originate from repositories.

Service Group (incl. Name-space if Existent)	Repository ID	Service Name (incl. Name-space if Existent)	Version
RSAU_LOG_API	SRVD_A2X	RSAU_LOG_API_SERVICE	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
Security Audit Log	Security Audit Log		<a href="#">Retrieving Security Audit Log [page 2777]</a>

## Service Response

The details included in the response vary according to the type of operation. For more information, see Operation for Security Audit Log Integration.

## Integration with SIEM Solution

Service name: RSAU\_LOG\_API

Communication scenario: SAP\_COM\_0750

This service enables you to retrieve the security audit log data. You can use the audit log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

## Additional Information

### ⓘ Note

For more details about Communication Management, see [Communication Management \[page 2591\]](#).

## 6.3.1.18.2.1.1 Operations for Security Audit Log Integration - Read

The Security Audit Log Integration - Read offers the following operations:

Operation	HTTP Method	Sample URL
<a href="#">Retrieving Security Audit Log [page 2777]</a>	GET	Example getting the first 1000 audit log entries:  GET <host>/sap/opu/odata4/sap/rsau_log_api/srvd_a2x/sap/rsau_log_api/0001/SecurityAuditLog?\$top=1000

## 6.3.1.18.2.1.1.1 Retrieving Security Audit Log

With `SecurityAuditLog` you can retrieve the Security Audit Log by using the HTTP method `GET`.

## Request

You have to include the following properties in the URL of the request:

Property	Necessity	Comment
eventID	Optional	Security Audit Log Event <ul style="list-style-type: none"><li>• String &lt;= 3 characters Example: AU1, AU2</li></ul>
log_tstamp	Optional	Timestamp of the security audit log event <ul style="list-style-type: none"><li>• String(\$date-time) Example: 2017-04-13T15:51:04Z</li></ul>
slgmand	Optional	Client within the SAP System <ul style="list-style-type: none"><li>• String &lt;=3 characters Example: 100</li></ul>
sid	Optional	System ID <ul style="list-style-type: none"><li>• String &lt;=3 characters</li><li>• Example: YI3</li></ul>
counter	Optional	Message counter <ul style="list-style-type: none"><li>• integer(\$int16)</li></ul>
slgtc	Optional	Transaction Code <ul style="list-style-type: none"><li>• String &lt;= 20 characters Example: VA01</li></ul>
slgrepna	Optional	Program Name <ul style="list-style-type: none"><li>• String &lt;= 40 characters Example: SAPMSSY8</li></ul>

## Response

The operation returns the Security Audit Log.

## Examples

### Get events within a specific timeframe:

```
GET <host>/sap/opu/odata4/sap/rsau_log_api/srvd_a2x/sap/rsau_log_api/0001/  
SecurityAuditLog?
```

```
$filter=(log_tstamp%20ge%202021-05-08T10%3A25%3A09Z%20and%20log_tstamp%20le%202021-05-09T10%3A25%3A09Z)
```

## Response

```
{
  "eventID": "AU1",
  "log_tstamp": "2021-05-08T10:26:22.740611Z",
  "slgmand": "100",
  "sid": "ABC",
  "counter": 0,
  "terminal_name": "",
  "user_fullname": "Example Administrator",
  "slgtc": "S000",
  "slgrepna": "RSBTCRTE",
  "rsau_text": "Logon successful (type=B, method=A)",
  "UserDescription": "Example Administrator"
},
{
  "eventID": "AU1",
  "log_tstamp": "2021-05-08T10:28:22.924215Z",
  "slgmand": "100",
  "sid": "ABC",
  "counter": 0,
  "terminal_name": "",
  "user_fullname": "Example Administrator",
  "slgtc": "S000",
  "slgrepna": "RSBTCRTE",
  "rsau_text": "Logon successful (type=B, method=A)",
  "UserDescription": "Example Administrator"
},
```

## 6.3.1.19 System Management

[Maintain User Sessions \[page 2779\]](#)

### 6.3.1.19.1 Maintain User Sessions

With this feature you can view all sessions containing locks in the current system. You can also display further information, such as associated business users. If required, you can delete a session, for example if it is blocking an app.

#### ⚠ Caution

Please note that it is not possible to delete sessions that are related to drafts.

## Key Features

You can use this app to:

- View details of the sessions, such as user IDs, user names, or number of locks
- Delete sessions if required

### Note

Please note that the business user does not get a warning message when a session is deleted. Therefore, we recommend that you do this with caution.

The following business catalog is required to make this app visible:

*System Management – Sessions (SAP\_CORE\_BC\_CST\_DP\_MAS\_PC)*

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SRV-APS-IAM.

### 6.3.1.20 Technical Monitoring

[Application System Overview \[page 2781\]](#)

[HANA Table Analysis \[page 2781\]](#)

[Sampled Work Process Data \[page 2782\]](#)

[HANA Thread Samples \[page 2783\]](#)

[System Workload \[page 2784\]](#)

[System Outbound Communication \[page 2784\]](#)

[Capture Request Statistics \[page 2785\]](#)

[Perform System Sizing \[page 2786\]](#)

[Health Monitoring \[page 2786\]](#)

[Schedule Metric Provider Collection \[page 2787\]](#)

[Partition HANA Tables \[page 2788\]](#)

[ABAP Runtime Errors \[page 2790\]](#)

[HANA Memory Analysis \[page 2790\]](#)

## 6.3.1.20.1 Application System Overview

**App ID:** F4031

With this app you can get access to a set of essential data to monitor the application system.

### Key Features

This app provides the following key features:

- Navigation to detail views, such as memory utilization, work process utilization, dialog step times, and top transactions by CPU utilization
- Thumbnail previews of essential application system metrics

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

### More Information

The application system overview is part of the technical monitoring cockpit. For more information about the technical monitoring cockpit, see the documentation at [https://help.sap.com/viewer/tmc\\_cloud](https://help.sap.com/viewer/tmc_cloud).

## 6.3.1.20.2 HANA Table Analysis

**App ID:** F4031

With this app you can get in-depth information about tables, for example, about their characteristics and resource consumption.

## Key Features

This app provides the following key features:

- View of the total memory consumption of the biggest tables
- Identification where you should optimize the tables and indexes in your system, for example, to save disk space
- For a selected table, detailed information about partitions, indexes, memory, and load

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

### 6.3.1.20.3 Sampled Work Process Data

**App ID:** F4031

With this app, you can check, for example, how much ABAP CPU time and main memory your work processes have consumed and find out the dialog and background work process utilization. This helps you to identify the top consumer workloads that might slow down your system and cause bottlenecks.

## Key Features

This app provides the following key features:

- Identification of resource utilization and main workload
- Analysis of concurrent workload
- Detail views of single work process samples
- Analysis of the root cause of an identified expensive workload or a bottleneck by further drilling down to single code lines via sampled stack traces

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-OP-MON.

### **6.3.1.20.4 HANA Thread Samples**

**App ID:** F4031

With this app, you can estimate how much HANA CPU time your custom applications consume. This helps you to identify your top workload that consumes most of the HANA CPU time, from where you can drill down to a detailed view of HANA thread samples and the related single SQL statements.

## **Key Features**

This app provides the following key features:

- Identification of main workload for HANA CPU utilization
- Detail views of single HANA thread samples
- Analysis of the root cause of expensive workload or resource bottlenecks by drilling down to the ABAP coding that triggered the database activity
- Navigation to the SQL statement that triggered the thread activity

## **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-OP-MON.

## 6.3.1.20.5 System Workload

**App ID:** F4031

With this app you can get an overview of your system workload and see how it behaves over time.

### Key Features

This app provides the following key features:

- Response time distribution in terms of, for example, CPU, database, and processing times
- Time series of the response time distribution of the system
- Drilldown to request processing details and ABAP statistics records

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

## 6.3.1.20.6 System Outbound Communication

**App ID:** F4031

With this app you can get an overview of the system outbound communication over time, which helps you identify obvious expensive outbound communication.

### Key Features

This app provides the following key features:

- It shows the top 10 communication scenarios by calling time.
- The *System Outbound Communication* screen and its related drill-down screens provide sample records of expensive system outbound communication.

- Outbound communication of type RFC, HTTP, and Web service is supported.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

### 6.3.1.20.7 Capture Request Statistics

**App ID:** F5699

With this app you can capture request statistics to find out which activities are running in your ABAP system. With the activation of a capture profile, you can capture system activities. Requests are recorded with technical information, such as response time, program name, or CPU time. Using the request statistics, you can investigate the performance of a system or of a tenant in more detail.

## Key Features

This app provides the following key features:

- Creation of capture profiles that define which request statistics are collected and when
- Display of profiles for capturing request statistics defined by SAP
- Activation and deactivation of profiles for capturing request statistics
- Navigation to the analysis of request statistics
- Activation and deactivation of SQL traces

## Supported Device Types

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-OP-MON.

### **6.3.1.20.8 Perform System Sizing**

**App ID:** F5700

With this app, you can calculate how many ABAP compute units are consumed based on captured request statistics of selected business scenarios. With the result of the system sizing, you can estimate how many ABAP compute units the production system will need.

#### **Key Features**

This app provides the following key features:

- Navigation from the definition of profiles for capturing request statistics to system sizing based on these statistics
- Estimation of required total ABAP memory, work process time, and ABAP CPU time in ABAP compute units for selected ABAP statistics records
- Fine-tuning of system sizing by changing the selected ABAP statistics records, the process executions per minute, or the available ABAP compute units

#### **Supported Device Types**

- Desktop
- Tablet

## **Component for Customer Incidents**

If you need support or experience issues, please report an incident under component BC-OP-MON.

### **6.3.1.20.9 Health Monitoring**

**App ID:** F6967

With this app you can check the health of your ABAP system. The KPIs and time series of various metrics are displayed on individual cards. These metrics indicate the health state of the system.

## Key Features

This app provides the following key features:

- Overview of the metrics for the monitored system  
Health metrics are represented by cards with a KPI that summarizes the current status.
- Display of the history of the metrics to, for example, compare trends, identify performance bottlenecks, or investigate resource shortages
- Navigation to related apps from some cards for further analysis or issue fixing

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

## 6.3.1.20.10 Schedule Metric Provider Collection

**App ID:** F7021

With this app you can schedule and monitor application jobs that collect metric providers that have been created in ABAP Development Tools (ADT) using ABAP for Cloud Development. After you've scheduled your application job and set up the communication to SAP Cloud ALM, your metrics will also be available in SAP Cloud ALM Health Monitoring.

## Key Features

This app provides a view on the [Application Jobs](#) app. Therefore, its key features are identical with the key features of the [Applications Jobs](#) app.

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

### Related Information

[Application Jobs \[page 2561\]](#)

## 6.3.1.20.11 Partition HANA Tables

**App ID:** F5466

With this app you can break down large columnar tables into partitions. By default, database tables on SAP HANA are not partitioned. The SAP HANA database allows for a maximum number of approximately two billion records per table. If this threshold is exceeded, your application stops working. To avoid this, you need to partition your tables in time.

### Key Features

This app provides the following key features:

- Search for large tables that can be partitioned according to certain filter criteria
- Partition large tables
- Undo the partitioning of a table by merging the partitions of a table back into one table

#### Note

With this app you can only partition your own tables or tables that have explicitly been classified for partitioning by SAP.

### Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

## More Information

For more information about the partitioning limits of the SAP HANA database, see SAP Note [2154870](#) and [Partitioning Limits](#).

### 6.3.1.20.11.1 Partitioning a Large HANA Table

Break down large columnar tables into partitions before the SAP HANA database threshold of two billion records per table is reached.

#### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for the *Partition HANA Tables* app.
2. To search for tables that contain a large number of records, enter your search criteria.

By default, the search criteria include the table name, the number of records, the total memory in MB and the number of partitions. If you want to include other search criteria, choose the *Adapt Filters* button.

3. Choose the *Go* button.

A list of tables that match the search criteria and that are eligible for partitioning is displayed, such as your own tables or tables released for partitioning by SAP. You can identify large tables in this list by checking the number of records and the total memory of the individual tables.

#### ⓘ Note

You can add or remove columns in the list by choosing the *Settings* button.

4. Choose the table that you want to partition and choose the *Partition Table* button.
5. In the *Change Partitioning Schema* dialog box, select the partitioning key and enter the number of partitions that you want to create. Then choose *Partition Table*.
6. In the *Confirm Table Partitioning* dialog box, choose *OK*.

A background job is scheduled on the backend system that carries out the table partitioning.

7. To check whether the job has finished and your table was partitioned successfully, refresh the list of tables by choosing the *Go* button.

#### ⓘ Note

You can always undo the partitioning of a table by choosing a partitioned table in the list and then choosing the *Merge Partitions* button. All partitions of this table will then be merged back into one partition.

## 6.3.1.20.12 ABAP Runtime Errors

**App ID:** F7770

With this app, you can display and analyze the ABAP runtime errors that have occurred in your current tenant.

### Key Features

This app provides the following key features:

- List of all ABAP runtime errors that have occurred in your current tenant, which you can personalize
- Display of summary information, the source code, and the call stack for each ABAP runtime error that has occurred
- Option to navigate to ABAP development tools for Eclipse (ADT) to fix the error or to display the error in the Web browser (HTML)

### Supported Device Types

- Desktop
- Tablet

### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

## 6.3.1.20.13 HANA Memory Analysis

**App ID:** F4031

With this app you can get an overview of the memory consumption of the HANA memory components over time.

### Key Features

This app provides the following key features:

- Identification of the components in your HANA memory that approach the critical memory limit that leads to bottlenecks in your HANA memory

- View of the total memory consumption of column store and row store tables with the option to navigate to detailed information for selected tables
- View of the content of the SQL plan cache with the option to analyze selected SQL statements
- View of the content of the buffer cache

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-OP-MON.

### 6.3.1.21 Manage Banks - Master Data

**App ID:** F6437

With this app, you can display and create master data for the banks your company, your business partners, customers, and your suppliers use to conduct business.

## Key Features

You can use this app to:

- Display banks and view their basic information, such as bank name, SWIFT/BIC code, bank number and address information.
- Create and edit both standard banks and internal banks.  
For more information about the bank categories the app supports, see [Banks and Bank Categories](#).
- Create and edit international address versions.  
For more information, see [International Address Versions](#).

#### ⓘ Note

Additional features relating to [Cash Management](#) can be accessed in app [Manage Banks – Cash Management](#). For more information, see [Manage Banks - Cash Mangement](#) under Feature Comparison for [Bank Management Apps](#).

### Note

This app contains in-app help for key fields and concepts. To display the help while working in the app, press F1 or choose the question mark displayed in the app header.

In addition, following technical features and options are supported:

- Banks can be replicated to other systems using SAP Master Data Integration. For more information, see [Master Data Replication Using SAP Master Data Integration \(Related Links\)](#).
- Banks can be imported from external files with other app [\*Import Bank Directories\*](#). For more information, see [Uploading Bank Data \(Related Links\)](#).

## Access Information

For information on what business role and business catalog need to be assigned to your user to access the app, see the overview table in the chapter [Applications for General Functions for Key Users](#).

## Supported Device Types

- Desktop
- Tablet

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component CA-BK-BNK ([Bank Master Data](#)).

## Related Information

[Master Data Replication Using SAP Master Data Integration](#)

[Uploading Bank Data with \*Import Bank Directories\*](#)

[Manage Banks - Cash Management](#)

## 6.3.1.22 Translation

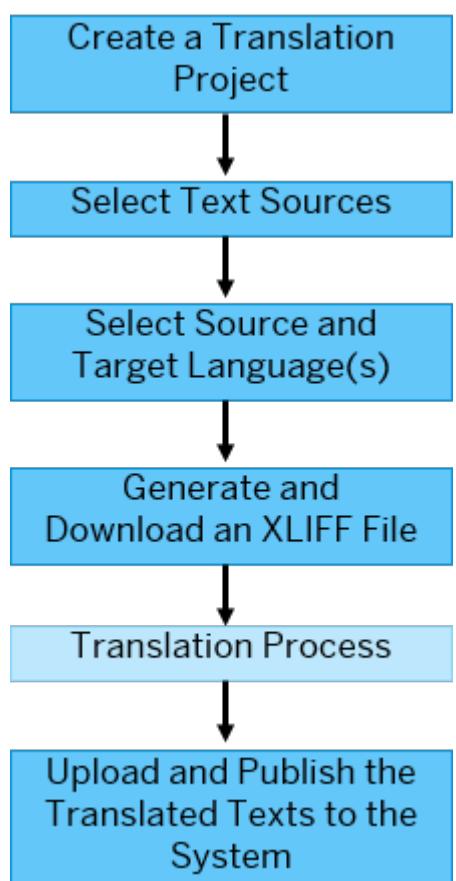
[Maintain Translations \[page 2793\]](#)

## 6.3.1.22.1 Maintain Translations

The *Maintain Translations* app provides an easy way to translate developer extensions into other languages.

### Context

You've built a developer extension and would now like to translate it into other languages. The *Maintain Translations* app allows you to select your text sources and generate an XLIFF (1.2) file containing the translatable texts as well as useful notes on them. XLIFF (XML Localization Interchange File Format) is a standard used in localization. The XLIFF file can then be forwarded to a dedicated translator or translation agency. Once the XLIFF file has been translated, you can use the *Maintain Translations* app to upload it and publish the translated texts into your system.



- [Create a Translation Project \[page 2794\]](#)
- [Select Your Text Sources \[page 2795\]](#)
- [Select Your Source and Target Language \[page 2797\]](#)
- [Generate and Download the XLIFF File \[page 2797\]](#)

- Upload and Publish the Translated XLIFF File to the System [page 2798]

## Key Features

You can use this app to:

- generate an XLIFF file of the texts that need to be translated.
- publish the translated texts to the system.

### Note

If you want to have your texts translated by SAP, you can use the SAP Translation Hub. For more information, see [SAP Translation Hub](#).

## Supported Device Types

- Desktop

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-DOC-TTL-MTR.

### 6.3.1.22.1.1 Create a Translation Project

#### Context

You want to translate your developer extension. Before you can select your text sources and target languages, you need to create a translation project.

### Note

Please assign the business catalog SAP\_A4C\_BC\_TRN\_MNG\_PC to ensure that changes can be recorded on transport requests.

#### Procedure

1. Open the *Maintain Translations* app.

2. Click **+** (Create) to create a new translation project.
3. Fill in an identifier for your project, a name and description, and the name of the package that should contain your translation project.
4. Click **Create** at the bottom right of your screen.
5. You now need to add your translation project to a transport. You can either use an existing transport request that matches the transport target of the package you have selected for the translation project, or create a new one by clicking **Create**. Save your changes by clicking **Select**.

## Results

Your project has now been created and added to the overview list.

## Related Information

[Select Your Text Sources \[page 2795\]](#)

### 6.3.1.22.1.2 Select Your Text Sources

#### Context

Select the text sources that you want to have translated.

#### Procedure

1. In the **Maintain Translations** app, open the project you want to add text sources to.
2. Scroll down to **Text Sources** and click **Add**.
3. Here you can select which text source types to add. The following text source types are supported:
  - Domains
  - Data Elements
  - Data Definitions (View Entities, Projection Views, Abstract Entities and Custom Entities)
  - Message Classes
  - Metadata Extensions
  - Application Log Objects
  - Business Configuration Objects
  - Text Pools
  - Text Tables

- IAM Business Catalogs
- CDS Type Definitions

It's also possible to specify a package from which all supported translatable elements will be pulled into the translation project. Please note that translatable elements in sub packages are not included.

4. Select *Add* to open the *Add Text Source* dialog which displays all text sources that can be added to the translation project. By filtering the type and name, you can choose the text sources you want to add. Select *Add* to add the text sources you selected to your translation project.

#### ⓘ Note

Text sources that can be translated by a translation project need to reside in the same software component as the translation project. The total amount of texts in a translation project must not exceed 1000. Be aware that a text source like a data definition might contain more than one text.

#### ⓘ Note

- If you want to test the translated texts of IAM business catalogs directly in the Fiori launchpad of the development system, you have to publish the IAM business catalog once again locally in ADT.
- If you want to test the translated texts of data definitions or metadata extensions directly in the development system with a Fiori app built on top of these objects, you should first reactivate the objects in ADT to ensure that all caches will be invalidated.

#### ⓘ Note

##### Text Tables

Text tables are database tables that were created as part of a business object and in which language-dependent texts are stored. The translation of text table entries is only available in environments that support transporting client-specific data.

A database table needs to fulfill the following requirements to be considered a text table by the *Maintain Translations* app:

- The database table has exactly one key field of ABAP Dictionary Built-in Type LANG.
- The database table is of delivery class C or S.
- The database table is located in the same software component as the translation project.
- The translation of text tables that contain key fields of type p (packed number), i (integer) or x (byte field) is not supported. That includes any kind of UUID and raw types, such as SYSUUID\_X16.

If these requirements are fulfilled, the database table can be added to a translation project as a text table. Additionally, the following rules apply:

- Every non-key field of the text table that is of an ABAP Dictionary built-in type that is character-like and has a minimum length of 1 will be interpreted as a text attribute for which translations can be maintained.
- When the translations are published, the rows containing the texts in the target language will be added to the selected transport via R3TR TABU.

In this tutorial ([Create an SAP Fiori-based Table Maintenance App with SAP BTP, ABAP Environment](#)) on how to create a business configuration application to maintain error code definitions, for instance, the language-dependent texts for the respective error code definitions are stored as entries in the database table ZERROR\_CODE\_T### with ### being replaced by a number chosen by the developer, such as 000. The non-key field for which translations can be maintained is field DESCRIPTION.

## Related Information

[Select Your Source and Target Language \[page 2797\]](#)

### 6.3.1.22.1.3 Select Your Source and Target Language

#### Context

Select your source language as well as the languages you want to translate your texts into.

#### Procedure

1. In the [Maintain Translations](#) app, open the project for which you want to select the source language and target language. You can choose to select either a single language or more than one language.
2. Scroll down to [Translations](#) and click [Create](#).
3. Select a [Source Language](#) and [Target Language](#) from the dropdown menus and confirm with [Create](#).

#### ⓘ Note

You can select from all of the languages that are installed in the system. In case of ABAP repository objects and text pools, it's only possible to maintain translations for languages other than the original language.

Note that deleting a previously created translation entry for a given source and target language pair will only delete the corresponding entry in the translation project. Already published texts for the target language of the translation will not be deleted from the system.

4. A new entry has now been added to the list.
5. Repeat these steps for all source and target languages.

### 6.3.1.22.1.4 Generate and Download the XLIFF File

#### Context

You can generate an XLIFF file for your source and target language(s) that you can download and send to your dedicated translator or translation agency.

## Procedure

1. In the [Maintain Translations](#) app, open the project you want to have translated and scroll down to [Translations](#).
2. Select the entry for the source and target language for which you want to generate the XLIFF file.
3. Navigate to [Download](#). Here you can select whether the XLIFF file should contain all texts ([All Texts](#)) or only those texts that were changed since you last published a translated XLIFF file in these languages ([Changed Texts](#)). If this is the first time you are generating an XLIFF file for the project in this target language, choose [All Texts](#).

The XLIFF file will now be downloaded to your computer. You can send it to your dedicated translator or translation agency to be translated.

## Related Information

[Upload and Publish the Translated XLIFF File to the System \[page 2798\]](#)

### 6.3.1.22.1.5 Upload and Publish the Translated XLIFF File to the System

## Context

Your texts have been translated and a new XLIFF file has been sent back to you. You can now upload and publish the translated texts into the system. Here's how:

## Procedure

1. In the [Maintain Translations](#) app, open your project and scroll down to [Translations](#).
2. Select the entry for the source and target language for which you want to upload and publish the translated XLIFF file.
3. Click [Upload](#).
4. Search for the translated file and confirm with [Upload](#).
5. Now the translations have been uploaded to the system. They still need to be published, however. Select the entry and click [Publish](#).
6. Select a transport to which you want to add the corresponding language (LANG) transport entries or R3TR TABU entries (in case of text tables) and click [Select](#).

Your translations have now been published.

## 6.3.1.23 Software Component Lifecycle Management

The software component lifecycle management allows you to manage the lifecycle of software components that are available in your ABAP environment landscape.

### Purpose

For the lifecycle management of software components you can use the app *Manage Software Components* in your launchpad that displays available software components, and imports them to service instances in your ABAP environment landscape. With the app, you can create new software components and delete them. Due to similarity to the Git protocol and its workflows, the following documentation uses the term **pull** as synonym for **import**.

One software component is comparable to a repository in Git. These "repositories" are centrally stored in separate units and cannot be referenced by other software components, which means the transport of development objects between the components is not possible. All software components are managed by SAP.

Once you have pulled a new software component to a service instance, a new structure package is created. The structure package name corresponds to the software component name. A software component itself is developed in ABAP packages in ABAP Developments Tools (ADT). The development objects are then uploaded to the structure package, and the software component is made available for the import to other service instances.

#### Note

In your system you will see all software components linked to the same global account that is the relevant layer for software component access control. Furthermore, only the global account is relevant for the visibility of the software components. Region or infrastructure provider do not matter. You only need to ensure that all systems are created within the same global account.

This is contrary to the overall SAP approach, where you can use the subaccount level inside a global account as a relevant access control layer (entity separation).

#### Caution

It is strongly discouraged to perform ABAP cloud development on a software component across multiple development systems. The reason for this is the potential for critical merge conflicts. When multiple developers work on the same software component simultaneously, edit it in different development systems, and release the transport requests afterward, inconsistencies and conflicts can arise in the target system.

This can lead to significant problems in the further development and maintenance of the software. It can even result in data loss. To avoid these issues, performing ABAP development on a software component in only one development system is highly recommended.

## Related Information

[Manage Software Components \[page 2805\]](#)

### 6.3.1.23.1 Supported ABAP Object Types

The following table contains all released ABAP object types that are supported in Git-enabled Change and Transport System (gCTS).

Object Name	Description	Generated
ADSO	DataStore Object	
ADVC	App Variant (LRep cross-client content)	
AIFC	Objects from AIF content	
AOBJ	Archiving Object	
APCO	Application Components	
APIS	API Release State of Objects	
APLO	Application Log Object	
AREA	InfoArea	
ATOM	ATO Manifest	
AUTH	Authorization Check Fields	
BDEF	Behavior Definition	
BRLC	Business Role Custom	
BUGR	Business User Group	
CDAT	View Cluster Maintenance: Data	
CDBO	Customer Data Browser Object	
CFDA	Custom Fields: BAdls of Business Context	
CFDF	Custom Field	
CHDO	Change Document	
CHKC	ATC Check Category	
CHKE	Exemptions	
CHKO	ATC Check Object	
CHKV	ATC Check Variant	
CLAS	Class	
CTRT	Currency Translation Type	
DCLS	ABAP Data Control Language Sources	

Object Name	Description	Generated
DDLS	Data Definition Language Source	
DDLX	Core Data Services Metadata Extensions	
DEVC	Package	
DMOD	Data Flow	
DOMA	Domain	
DPTA	Data Protection Template	
DRAS	CDS Aspects	
DRTY	Dictionary: CDS Type Definitions	
DSFD	CDS Scalar Function Definition	
DSFI	CDS Scalar Function Implementation Reference	
DTDC	Dictionary Dynamic Cache	
DTEB	Dictionary Tuning: Entities Buffer	
DTEL	Data Element	
DTPA	Data Transfer Process: Active Version	
EEEC	Enterprise Event Enablement - Event Consumer	
ENHO	Enhancement Implementation Object	
ENHS	Enhancement Spot Object	
ENQU	Lock Object	
ESHA	Enterprise Search: DDL Activation	
EVTB	Event Binding	
FUGR	Function Group	
FUNC	Function Module	
G4BA	OData v4 – BEP - Service Group and Assignments	
G4BS	OData v4 – BEP - Service	
GCPM	SAP Gateway OData Client Proxy - Proxy Model	
GSMP	Metric Provider	
HCPR	SAP HANA Composite Provider	
HTTP	HTTP Service	
INA1	InA Service	
INTF	Interface	
IOBJ	InfoObject	

Object Name	Description	Generated
IWMO	SAP Gateway Business Suite Enablement - Model	
IWOM	SAP Gateway: Model Metadata	Yes
IWSG	SAP Gateway: Service Groups Meta-data	Yes
IWSV	SAP Gateway Business Suite Enablement - Service	Yes
IWVB	SAP Gateway Business Suite Enablement - Vocabulary Annotation	Yes
LRCC	LRepository cross-client content	
LRCD	LRepository client-dependent content	
LRST	LRepository client-dependent content - customizing requests	
MSAG	Message Class	
NONT	SAP Object Note Type	
NROB	Number Range Object	
NTTY	Note Type	
OA2S	OAuth2 Scope Yes	
PCFN	Predefined Field: Extensible Node	
PROG	Program	
RABR	Read Access Logging: Recording	
RADB	Read Access Logging: Dynpro	
RAGB	Read Access Logging: Generic Channel	
RARB	Read Access Logging: Remote Function Call	
RASB	Read Access Logging: Web-Service	
RAWB	Read Access Logging: Web Dynpro	
RDOM	Read Access Logging: Log Domain	
RGPR	Business Role Group	
RONT	SAP Object Type	
RPUR	Read Access Logging: Purpose	
RSDS	DataSource in BW	
RSPC	Process chain	
RSPI	Interrupt Process	
RSPT	Trigger Process	
RSPV	Process variants	
SAJC	Application Job Catalog Entry	

Object Name	Description	Generated
SAJT	Application Job Template	
SCO1	Communication: Communication Scenario	
SCO2	Communication: Inbound Service	Yes
SCO3	Outbound Service	
SIA1	IAM: Business Catalog	
SIA2	IAM: Restriction Type	
SIA3	IAM: Authorization Object Extension	
SIA5	IAM: Restriction Field	
SIA6	IAM: App	
SIA7	IAM: Business Catalog App Assignment	Yes
SIA8	IAM: Business Role Template	
SIA9	IAM: Business Role Templ. Bus. Catalog Assignment	
SIAD	Business Role Template Fiori Space Assignment	
SICF	ICFService	
SKTD	Knowledge Transfer Document	
SMBC	Maintainable Business Configuration	
SMIM	MIME Object	Yes
SOD1	API Package	
SOD2	API Package Assignment	
SPRV	SOAP Provider Model	
SUSI	Assignment: Service -> Authorization Objects (with TADIR)	
SPRX	Proxy Object	Yes
SQL1	ABAP SQL Service	
SRVB	Service Binding	
SRVC	Service Consumption Model	
SRVD	Service Definition	
SUSH	Assignment: Service --> Authorization Objects	
SUSO	Authorization Object	
TABL	Table Definition	
TABU	Table Contents	
TDAT	Customizing: Table Contents	

Object Name	Description	Generated
TOBJ	Definition of a Maintenance and Transport Object	
TRCS	Communication Structure for Transformation	
TRFN	Transformation	
TRPC	Translation Project - Customizing	
TRPR	Translation Project	
TTYP	Table Type	
TYPE	Type Group	
UIAD	Application Descriptors for the Fiori Launchpad Environment	
UIPC	Fiori Launchpad Page - Customizing	
UISC	Fiori Launchpad Space - Customizing	
UOMT	Quantity Conversion Type	
VDAT	View Maintenance: Data	
WAPA	BSP Page	Yes
WEBI	Service definition - WEBI	Yes
XINX	Ext. Index	Yes
XSLT	Transformation	

### ① Note

Object types flagged as "generated" cannot be created by customers directly but they are secondary objects that are necessary for the functionality of some objects.

### Customizing

Object Name	Description
CDAT	View Cluster Maintenance: Data
SNUMS	Number Range Configuration
TABU	Table Entries
TDAT	Customizing: Table Contents
VDAT	View Maintenance: Data

## 6.3.1.23.2 Manage Software Components

You can use this app to create, display, clone, delete and configurate software components in your ABAP environment landscape. Moreover, you can pull (import) changes from the central software component into other instances.

### Purpose

This app allows you to create a software component on a service instance and make with a 'Clone' and import changes (=commits = transport requests) on other instances after an initial clone using a 'Pull'.

### Key Features

You can use this app to:

- display available software components
- create new software components
- pull software components and display pulls
- delete software components
- display logs for performed actions like Clone, Pull, and other
- create and delete branches
- display pushed commits (= released transport request)
- create and delete tags on commits
- configure the repository role (Source or Target)
- enable or disable a rollback mechanism for failed pull operations

In addition, the app supports the integration with Git.

### Supported Device Types

- Desktop
- Tablet
- Smartphone

### Related Information

[How to Display Available Software Components \[page 2806\]](#)

[How to Create Software Components \[page 2807\]](#)

[How to Pull Software Components \[page 2810\]](#)  
[How to Delete Software Components \[page 2812\]](#)

## 6.3.1.23.2.1 How to Display Available Software Components

### Context

You want to see which software components already exist in the environment and are locally available. To display available software components, perform the following steps:

### Procedure

1. After opening the app, choose *Go* on the *Manage Software Components* screen to display the list of available software components. The list provides you with the following information:

Column Name	Description
<Name>	Name of the software component that has to be unique per service instance. The maximum length of the name is restricted to 18 characters including a namespace.
<Description>	Description of the software component, which can only contain 60 characters.
<Type>	Type of the software component. The software component must either be <i>Development</i> or <i>Business Configuration</i> .
<Checked Out Branch>	The name of the branch that is checked out.
<Cloned>	Shows whether or not the software component has been cloned.
<Created On>	Date and time when the software component was created.
<Created By>	Email address of the technical user who created the software component.
<Changed On>	Date and time when last changes were made to the software component.

Column Name	Description
<Changed By>	Email address of the technical user who performed the last changes on the software component.

2. To display detailed information on a software component, choose the list entry.

## 6.3.1.23.2.2 How to Create Software Components

### Context

To create a new software component, perform the following steps:

### Procedure

1. On the *Manage Software Components* screen, select *Create*.
2. In the *New Software Component* popup, select your namespace from the drop-down menu and enter a name for the software component. Optionally you can also enter a description (max. 60 characters), which explains the functions of the component and is displayed in the app. Choose a type for your software component: Select either *Development* or *Business Configuration* from the drop-down menu.

Software Component Types

Software Component Type	Description
Development	Used for standard ABAP application development.
Business Configuration	Used to transport customizing content from one ABAP environment system to another ABAP environment system.

#### ⓘ Note

You can create multiple business configuration software components. You can only clone one business configuration software component to a system. You can only release customizing transports to a business configuration software component

3. Click *Save* to make the software component centrally available.
4. Back on the *Manage Software Component Lifecycle* screen, choose *Go* to display the new software component in the list.

1. As soon as your software component is cloned into a service instance, a structure package with the same name as the software component is created there. The structure package can contain several ABAP packages. See [How to Clone Software Components \[page 2808\]](#).
2. The instance is integrated with the ABAP Development Tools (ADT), where ABAP development objects are created in development packages and uploaded to the structure package.

**ⓘ Note**

Note that you have to create a development package first. Afterwards, create ABAP development objects in this newly created development package. The development package is a sub-package of the structure package dependent on the software component.

**→ Tip**

We recommend that you add all your cloned software components (structure packages) to your [Favorite Packages](#) in eclipse to have an easy and quick access to it.

3. After the transport of the structure package, the software component becomes available for all instances and can be cloned to other system instances.  
As soon as the structure package is transported, for example, to provide a new version of the software component, the date and time of the transport as well as the user who triggered the transport are displayed in the [Branching](#) section of the detail page of a software component. More information about the released content can be found when navigating on the used branch.

**ⓘ Note**

Note that a maximum of 99 software components can be created per customer.

### 6.3.1.23.2.3 How to Clone Software Components

#### Context

If a software component is not locally in your system yet, you will first need to clone it once to import it into your system.

You can now deactivate the rollback mechanism for clone actions in the Manage Software Components Fiori App. The clone rollback should only be deactivated in exceptional cases.

Example of a use case: If you have a faulty remote repository (software component) but no development system on which the repository (software component) is still available locally. Then you need a way to clone the faulty software component and fix the errors.

**ⓘ Note**

Cloning a software component is only necessary once per system instance.

## Procedure

1. Go to the details page of the software component you would like to clone.
2. Click [Clone](#).
3. A pop-up opens. You can now select which branch you would like to import. If the software component does not have branches yet, the master or main branch (dependent on when the repository was created) is automatically selected. Select [By Tag](#) if you want to clone a specific git tag, which has been assigned to a commit. Select [By Commit](#) if you want to clone a software component with a specific commit ID. Confirm by clicking [Clone](#).

### Note

There is no input validation for the entered values for the options [By Tag](#) and [By Commit](#). If the entered values are not valid the cloning process will start, but will be shown as failed in the end of the cloning process. In this case, the clone [Latest](#) option is automatically called as a fallback. With this, the repository is still useable, but not in the originally requested version.

### Note

If the initial clone of a software component runs into errors, the partly imported objects will be removed again from the system instance and the system will be kept clean. The clone rollback is now available for all system types.

4. The software component with the selected branch is now imported into your current system instance. The [Clone](#) button turns into a [Pull](#) button. From now on, you can use it to pull (remotely available) changes of your software component to your system.

### Note

It is recommended to clone multiple software components one by one. Please clone software components with objects that are reused by other components first.

## Related Information

[How to Pull Software Components \[page 2810\]](#)

## 6.3.1.23.2.4 How to Pull Software Components

### Context

You can pull (remotely available) changes of your software component to the service instance you are currently on. To do the pull, perform the following steps:

#### ⓘ Note

Keep in mind that if a software component is not in your system yet, you will first need to clone it once. For more information see [How to Clone Software Components \[page 2808\]](#).

### Procedure

1. Navigate to the detail page of a software component from the list and choose *Pull* in the page header.
2. Select how you want to pull the software component. By selecting *Latest*, the last remote commit will be pulled into the system. Select *By Tag* if you want to pull a specific git tag, which has been assigned to a commit. Select *By Commit* if you want to pull a software component with a specific Commit ID.

#### ⓘ Note

To pull a specific tag, at least one tag has to exist on a commit on the currently active branch.

#### ⓘ Note

A rollback mechanism is activated which allows an automatic roll back to the last valid commit of the repository, in case an invalid commit is pulled. We define an invalid commit as a commit that can cause import errors during a pull operation. The pull of invalid commits is allowed for development and test systems only. If only one out of multiple commits causes an issue, none of the commits will be pulled.

Nevertheless, this mechanism cannot reverse any data loss that may occur due to ABAP Dictionary changes. For example, in the case of field shortening of a database table, the field length can be rolled back, but the lost content remains irretrievable. Therefore, ensure that deployments to the production systems are checked in the advance (via CI/CD processes), for example on quality systems.

3. Display the list of all pulls, checkouts and other actions by navigating to the *History* tab.  
To display pulls and checkouts that have been triggered for a particular software component, first select the component from the list and then scroll down to the *History* section where all actions are listed in the *Recent Actions* tables. In both cases, you will be navigated to the *Software Component Details* screen.
4. To display or refresh the list of *Recent Actions*, choose *Go* or click on the refresh button inside the search bar.

#### Field Description

Column Name	Description
<ID>	A unique key (UUID) of the software component. This column is initially hidden.
<Branch Name>	The branch which was used for the executed action.
<Short Commit ID>	The short commit id that was used for the executed action.
<Tag>	The name of the tag which is eventually assigned to the used commit.
<Last Status Change>	The relative time from current system time and the timestamp of the last status change.
<Status>	Displays whether the pull or checkout process has been successful, is still running, or there has been an error.
<p><b>ⓘ Note</b></p> <p>Only one pull can be in status <i>Running</i>. If you trigger several pulls, other software components will switch to status <i>Error</i>.</p> <p>If an error has occurred during a pull, you need to click Pull again.</p>	
<Started By>	E-Mail address of the user who started the pull.
<Start Time>	Date and time when the pull has been triggered.
<Change Time>	Date and time of the changes during the pull procedure.

#### ⓘ Note

The <ID>, which is a unique key (UUID) of the software component is not in the table, but visible in the action detail page.

5. You can display the pull details by choosing the entry from the list. The *Execution Log* and *Log Overview* are displayed as a table. You can find the new tables when navigating to the detail page of every action entry. Additionally, every log line is classified with a criticality level. A logline can have the criticality level "Information", "Success", "Warning" or "Error". Each level is represented with a colored criticality icon. Furthermore, you can also download the logs as an Excel-file. This may be helpful when communicating with SAP in the context of error analysis.

### Note

Note that the execution logs and transport logs are deleted periodically. Please use the download feature in the UI or the OData service `MANAGE_GIT_REPOSITORY` (see [Pulling Git Repositories to an ABAP Environment System \[page 788\]](#)) to download the logs.

#### Field Description

Column Name	Description
<code>&lt;Index&gt;</code>	Index number to sort the table properly.
<code>&lt;Log&gt;</code>	Name of the log.
<code>&lt;Severity&gt;</code>	Can have the status success, warning, error or other.
<code>&lt;Transport Request&gt;</code>	Transport request ID which was used for the log.
<code>&lt;File including Path&gt;</code>	The system file path where the log files are saved.
<code>&lt;Timestamp&gt;</code>	Date and time when the log was created.

## Related Information

[How to Clone Software Components \[page 2808\]](#)

[How to Configure Software Components \[page 2818\]](#)

## 6.3.1.23.2.5 How to Delete Software Components

### Context

You want to delete a software component in order to:

- get rid of software components that are no longer used.
- get rid of software components that have been created by mistake (such as a typo in the software component name).

For every delete activity, the executing business user is documented in a log.

## Procedure

To delete a software component, perform the following steps:

1. In the *Manage Software Components* app, pick the software component that you want to delete out of the list.
2. Click onto the software component to be redirected to its detail page.
3. Choose the *Delete* button in the upper right corner.
4. You will be prompted to confirm the action and are able to choose if you want to delete only the local repository or the local repository in combination with the remote repository.

## Software Component Deletion in a Service Instance

### Types of Software Component Deletion

Empty Software Component	Used Software Component
The software component contains no object locally in the service instance and no changes are committed to the remote repository. The remote repository, the associated structure package of the software component, and the software component itself are deleted. The deletion of the ABAP structure package and software component is executed without creating a transport request.	The software component contains objects locally and possible changes are committed to the remote repository. The remote repository is then deleted and the software component is locked. Since the software component is locked, no object changes can be released (pushed to remote) anymore.

### Local and Remote Deletion - SAP-managed Software Components

Use Case	Local Deletion	Local and Remote Deletion
	Only the local representation of the software component is deleted. Local means all ABAP objects including the parent structure package.	When deleting local and remote representations, local means only the current system instance is deleted.  <i>The software component will be remotely deleted. As a consequence, no actions can be performed anymore.</i>  <i>Please make sure to first delete the software component locally on all system instances where it's cloned. The component can't be deleted from the other systems after it's remotely erased. This action can't be undone.</i>

<b>Consequences</b>	You can re-clone the software component again.	This step should be performed as a last step once all local deletions on other system instances have been performed.
	<p><b>⚠ Caution</b></p> <p>A local deletion can be critical when database tables are deleted. The local (not transported) table entries contained in this table will be lost. This is a possible scenario in test or production systems, since development systems usually don't contain any business data (like table entries).</p>	<p><b>ⓘ Note</b></p> <p>A distributed local deletion (deleting local software components on other system instances than the current one) is not possible.</p>

## Local and Remote Deletion - Customer-managed Software Components

	<b>Local Deletion</b>	<b>Local and Remote Deletion</b>
<b>Use Case</b>	Only the local representation of the software component is deleted. Local means all ABAP objects including the parent structure package.	The remote deletion of the software component is performed if the checkbox <i>Unregister Repository</i> was selected. This option will unregister the linked git repository from the <i>Manage Software Components</i> app, in addition to deleting the software component locally. If this option is selected, the linked Git repository can't be used in relation to the software component again, and it can't be linked with new software components. This action is irreversible.
<b>Consequences</b>	You can re-clone the software component again.	This step should be performed once all deletions on other system instances have been performed and only intended for cases when the remote Git repository is no longer needed.

## Resulting behavior in service instances where the deleted software component is still imported

1. No further Git actions, including pull, switch branch and others, are possible.
2. Changes to SAP objects of the deleted software component can no longer be released.
3. Non-released requests, in which object changes have already been recorded before the software component was deleted, must be handled specially.

For example, you can delete the recorded object changes from the transport request. Alternatively, the transport layer of the transport request can be deleted.

## Result

The selected software component is deleted centrally if the checkbox *remote deletion* was selected. Otherwise it is only a local deletion, affecting the current system instance.

### ⚠ Caution

Here, centrally means the repository visible throughout the global account is deleted.

You can no longer pull or clone the software component to a service instance.

If this software component has already been cloned to a service instance, objects are not deleted in the ABAP instances but the status of the software component and all belonging objects are changed to *read-only*. You're not able to make any changes to the dedicated structure package and the development package that is contained in there. If you want to delete the objects, you must delete them and import the deletion into other systems, before you delete the software component.

## Restoring Deleted Software Components

### ⓘ Note

Currently, you can't restore software components that have been deleted. Use new software component names instead of reusing previously deleted software components or software component names.

## 6.3.1.23.2.6 How to Configure Your Git Repository

Up until this point, the ABAP repositories in SAP BTP, ABAP environment were managed by the Manage Software Components Fiori app, which was the only interface where users could see the status of their remote repository. Bring Your Own Git (BYOG) allows users to connect their own Git repositories with the Manage Software Components app. In doing so, the user has a better overview of their own code, branches, and commits. However, the user is not allowed to make code changes in their remote Git repository and import these changes into the local SAP BTP, ABAP environment system instance, as this might violate any guidelines for ABAP cloud development. See [Supported ABAP Object Types](#)

## Prerequisites

- You have created a new repository with the preferred Git provider. The repository does not have any objects except for a readme file.

- You have created a branch in your repository before cloning it.
- Your repository is publicly accessible on the internet. Repositories hosted on a private company network are currently not supported.
- You have administrative authorizations to push, pull and clone from the repository. Credentials must include a username and a password or token.
- The developer users have administrative authorizations to the Manage Software Components app.
- Production, development and/or test system instances are in the same global account.
- The Git provider must trust a (root) certificate that is available in the SAP standard certificate trust list. Please check the current trust list [here](#) and compare it with your Git provider certificate.

## Restrictions

- Moving a BYOG software component from one global account to another is not supported.
- Changing the URL of the remote repository after it has been linked to a software component is not allowed. After changing the URL, the repository can no longer be used.
- It is important to distinguish between the user (pusher), whose credentials are entered during the clone process, and any other ABAP users (committers) that develop and release objects to be pushed to the remote repository. Although the committer will be noted in the commit history, the release and later the push will technically be performed using the credentials of the pusher. Therefore, it is recommended that the pusher does have push, pull and clone authorizations.

## Procedure

Follow the steps below to create and manage a BYOG (Bring Your Own Git) repository in the Manage Software Components (MSC) app.

### Creating a remote repository

To create a BYOG repository, first create a new repository in your preferred Git provider.

Make sure to create a branch as well, as a default branch is necessary for the clone later on.

Next, copy the HTTPS link (must end with .git) to clone the repository in the Manage Software Components app.

### Create a Software Component and link the remote repository

In the Manage Software Components app, create a new software component and enter the URL from the previous step in the *Repository URL* field. Select the correct Git provider for your repository. The currently supported providers are GitHub, GitLab, Azure and Bitbucket. Choose '*Git (others)*' if none of the other options apply.

#### Note

When specifying the repository URL, please note that it must be case-insensitive and unique. This prevents multiple software components from being linked to a single or identical Git repository.

Add a name for your software component and choose the component type as development. Afterwards, click on [Create](#). The software component is linked to the remote repository and is now ready to be cloned into the system.

## Cloning the Software Component

To clone the software component, click on the [Clone](#) button on the top right corner. A dialog will pop up:

Enter your user credentials that you would normally use to clone the repository, respectively a username and a password or token.

### ⓘ Note

Remember to enter the credentials of a user that has administrative rights to the repository and has authorizations to push and pull changes directly into the repository.

Select the radio button which describes your method of authentication, either a password or an OAuth token. Then click on [Next](#). If the credentials are correct, the next step [Branch Selection](#), will be displayed. Otherwise, re-enter the correct credentials before you move forward.

From the dropdown select the name of the branch you want to clone. Click on [Next](#).

Select the repository role and import options. Choose [Latest](#) to import the latest commit.

Click on [Clone](#). When the clone is finished, the status should be green.

After the clone is finished, notice that there will be a new commit [Modify repository layout](#) generated in the remote repository by commit author `noreply@example.com` and a newly added file. This file is necessary for pushing and pulling changes into the SAP system.

### ⚠ Caution

Do not edit or delete this file at any given time.

## Creating Branches

To create a branch, you should do so using the native interface (e.g., Git interface) of your git provider linked to your repository.

In the MSC app simply refresh the branch table to see the new branch. You can proceed as usual with the checkout.

### ⓘ Note

To be able to work with an alternative branch, you should create the branch after the initial clone. It is important that any new branches are created based on the [Modify repository layout](#) commit, as mentioned above. The file generated from this commit will allow import into the local system instance.

## Deleting a Software Component

Deleting a software component in the [Manage Software Components](#) app with the default settings will only delete the local representation of the software component in the ABAP system instance. Releasing transport requests will no longer be possible.

Selecting the [Unregister Repository](#) option will additionally unlink the remote repository from the software component and delete the software component completely from your global account.

Deleting the software component is an irreversible action. To delete the component, simply proceed by clicking **Delete** in the dialog. The remote Git repository will not be deleted, but it cannot be used again and be linked to new software components. This action is irreversible, therefore ensure to first delete the software component on all system instances where it is cloned.

### Creating and deleting tags

Tagging functionality is the same as in classic non-BYOG software components. To create a tag, go to the commit view of your selected branch. The created tag will also be visible on your remote repository. Deleting a tag will also delete the tag in the remote repository.

## Related Information

[Development in the ABAP Environment](#)

## 6.3.1.23.2.7 How to Configure Software Components

### Context

You can configurate software components in the *Repository Settings*, which can be opened by clicking on the *Settings* button.

### Key Features

You can do the following adjustments to the selected software component:

- Repository Role:  
Use the repository role to configure how transport requests behave when they're released. If the option *Target* is selected, the transport requests will not be pushed to the remote repository. However, if the *Source* option is selected, the transport request can be pushed to the remote repository.
- Rollback Mechanism:  
This rollback mechanism can be enabled and disabled. If enabled, only valid commits can be imported in the repository. If disabled, all commits can be imported even if they are invalid. For more information about this mechanism, see: [How to Pull Software Components \[page 2810\]](#).
- Update Credentials (BYOG software components only):  
Use this option to update your Git user credentials that were saved when the software component was cloned. This option is enabled only for Bring Your Own Git components that have already been linked to a Git repository. For more information on that, see [How to Configure Your Git Repository \[page 2815\]](#).

## 6.3.1.23.2.8 How to Export Using SAP Cloud Transport Management

### Context

1. A developer creates and modifies ABAP coding in the SAP BTP, ABAP environment by using the Eclipse-based IDE (ABAP development tools for Eclipse) with a rich debugger, troubleshooting, and testing tool support.
2. The changes are recorded in an ABAP transport request that is finally released. During the release, the changes are committed and pushed to the configured development branch of the corresponding Git repository (there is a one-to-one relation between an ABAP software component and a Git repository).
3. After testing and after the release decision, a release branch is created from the main branch.
4. The resulting combination of the corresponding software component and either the commit ID, Git tag, or the branch name is used as a reference by the following export call to the cloud transport management service (cTMS). It is recommended using the commit ID, because the latest commit of a branch may change, resulting in unrepeatable deployments.
5. In cTMS a transport request is created containing the reference and added to the import queue of the following cTMS node (given by the configured service transport route).
6. Finally, a transport administrator uses the cloud transport management service UI to generate the import to the SAP BTP, ABAP system configured on the cTMS transport route. In doing so, the service calls an import API of the target SAP BTP, ABAP environment system by passing the reference.
7. The import is done asynchronously, and the progress status is monitored by the service. When the import is finished, the service receives the transport logs from the target SAP BTP, ABAP environment instance.

### Procedure

Before we can start releasing transport requests that generate commits, a new communication arrangement is needed in order to use an endpoint.

1. Create a service instance and a service key of SAP Cloud Transport Management service. For more information, see [Creating a Service Instance and a Service Key](#).
2. Navigate to the [Communication Arrangements](#) Fiori app.
3. Click on [New](#) and search for SAP\_COM\_0599 communication scenario in the list. Type in the name. Click on [Create](#).
4. On the following screen, a new communication system should be created. Click on [New](#). Type in your System ID, which is the URI from your service key. Click on [Create](#).
5. On the communication system creation page find general properties and define the [Host Name](#) parameter according to the URI property in the service key created for SAP Cloud Transport Management service. For example <https://transport-service-app-backend.ts.cfapps.eu10.hana.ondemand.com>
6. Then find [OAuth 2.0 Settings](#) and put in a [Token Endpoint](#), which is the URL from your service key. For example [tmststest.authentication.sap.hana.ondemand.com/oauth/token?grant\\_type=client\\_credentials](https://tmststest.authentication.sap.hana.ondemand.com/oauth/token?grant_type=client_credentials)
7. Now it's time to create a user for outbound communication. Click on [+](#) and select oAuth 2.0 from the authentication method dropdown.

- Type in your OAuth 2.0 client ID and client secret that can be taken from the service key for the basic authentication. Click on [Create](#).
  - Define a cTMS node name that you will need for the export. The selected transport node must allow uploads and you should use the same name when creating the communication arrangement and creating the same in cTMS.
- For more information, see [Create Transport Nodes](#). Click [Save](#) afterwards to complete the process.

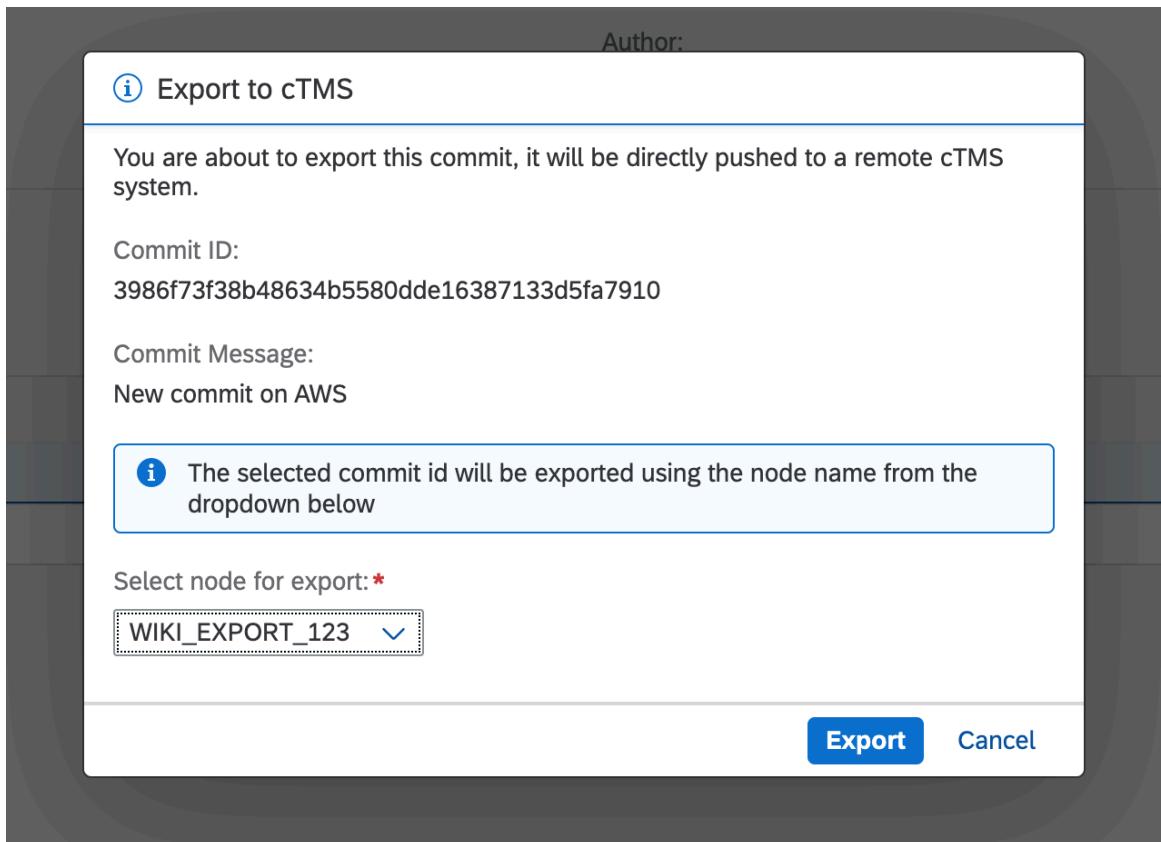
## Export to SAP Cloud Transport Management using the Manage Software Components app

- Open the *Manage Software Components* app and navigate to the desired software component and a branch.
- Select a commit for the export and click on the [Export Commit](#) button.

### List of Commits

Commits (2)   Standard ▾			Search	Add Tag	Export Commit	≡	✖	✖	
Pulled	Short Commit ID	Message							
<input checked="" type="radio"/> Yes	386dbfc7	[H01K900026] CSW Demo							>
		Pushed to Remote: Yesterday							
<input type="radio"/>	✓ Yes	e0c22f95	Initial repository commit						>
		Pushed to Remote: Yesterday							

- A new dialog will open, where you will be prompted to select a cTMS node you want to use for this export.



- Click on [Export](#) after you are ready with the selection. A status modal in the center of the screen will let you know if the process was completed successfully.

5. In case of an error, you will get an HTTP error code with a description.

## Related Information

[Set Up the Environment to Transport Content Archives directly in an Application](#)

[Configuring the Landscape](#)

[Using the Import Queue](#)

[Creating Destinations for Deployment of References of SAP BTP, ABAP environment](#)

### 6.3.1.23.2.9 How to Work with Branches

#### Context:

To allow you to easily work on and switch between different versions of code, you can create branches, a copy of your existing code at a certain point in time, that you can check out and work on without changing your original code.

Find out how to create, check out and switch between different branches:

#### Procedure:

1. Upon opening the **Manage Software Components** app, you'll see a list of all available software components. Select the one you would like to work on.
2. Scroll down to [Branching](#) to see a list of all available branches for your component. The table shows you which branch the different branches originally derived from and which branch is currently checked out. The first branch is always named 'main'. The main branch will be created directly with the creation of the software component. If a branch was successfully created, it should directly be visible on all other system instances.  
The column [Delta](#) shows information about the currently active/checked-out branch. It displays how many commits the branch is behind the remote repository (1 commit behind remote) or just informs, that the branch is up to date (No new commits). This information is only valid for the currently active branch.  
If the branch is not active, the column value will be set to "Please checkout first". If the branch is multiple commits behind, the column value will be set to "X commits behind remote", where X is the number of new commits on the remote branch.
3. Note that you need to clone the repository once before you can check out a branch. Clone the repository by clicking [Clone](#).
4. To create a new branch, click on [Create](#). Choose a name for your new branch and select the parent branch from the drop down menu. Confirm with [Create](#).

You can also create a branch based on a specific commit state. This referenced commit must be available in the current system, i.e. included in the currently selected branch. To do this, simply fill in the *Commit ID* in the *Create New Branch* popup. The commit ID can be provided in short or long format. Note that short commit IDs should only be used if the software component has already been pulled. Otherwise, the long ID needs to be used.

#### ⓘ Note

When you delete a branch and recreate it with the same name, the creation will be denied with an error popup saying "Branch cannot be recreated". As a consequence, a branch name has to be unique once it has been used. Please choose another name in that case.

5. Refresh the page. Your new branch has now been created and added to the table.
6. If you want to work on the new branch, you can check it out by selecting it and clicking *Checkout*. All objects in the target branch will overwrite existing objects in the system without further notice. It is therefore important to ensure that all open transport requests in the current branch are released before switching to a new branch.  
A new checkout is not possible while a checkout or pull is already in progress.
7. The branch selected for checkout is now available in the system.
8. You can easily switch between all your different branches by using the *Checkout* button.

#### ⓘ Note

##### **Branch Deletion:**

While it is possible to delete branches, please consider the possible negative effects of branch deletion, like losing the commit history.

Please make sure that the branch you are deleting is no longer in use on other systems. If you delete the branch while it's in use in other systems, this code can no longer be released there, even if there are open transport requests.

The branch will be deleted locally and also on the remote repository. After the branch deletion, a pull of this branch is no longer possible on all system instances.

#### ⓘ Note

The merging of branches is currently not supported.

## 6.3.1.23.2.9.1 How to Tag Commits

You can assign a tag name and tag description to individual commits.

1. In the *Manage Software Components* app, open your software component.
2. Navigate to a branch. Commits on this branch are displayed in the *List of Commits*.
3. Select a commit and click *Add Tag* to add a tag name and a tag description.
4. **(Optional)** Navigate to the *Tagging* tab of your software component to see a table listing all the tags on the branch that is currently checked out/active. Log entries for the creation and deletion of tags are also displayed in the *Recent Actions* table in the *History* tab.

### Note

Note that a commit can have multiple tags assigned to it. There is no restriction on how many tags a commit can contain.

## 6.3.1.23.3 API for Managing Software Components

You can use the MANAGE\_SOFTWARE\_COMPONENTS API to work with Software Components on SAP BTP, ABAP Environment systems. The MANAGE\_SOFTWARE\_COMPONENTS API belongs to the Communication Scenario SAP\_COM\_0948 and is an integral part of the partner add-on build pipeline.

### 6.3.1.23.3.1 Cloning Software Components to an ABAP Environment System

- You have an ABAP environment system.
- You have created a communication user as described in [How to Create Communication Users](#).
- You've created a communication system as described in [How to Create Communication Systems](#).
- You've created a communication arrangement as described in [How to Create a Communication Arrangement](#).
- You have selected the communication scenario SAP\_COM\_0948 for your communication arrangement and have mapped it to your communication system.

If a software component is not in your system yet, you will first need to clone it once to import it into your system. Please note that you can clone a software component only once into your system.

#### 1. *Authentication on the server*

The first step serves the authentication on the server. The response header contains an `x-csrf-token`, which is used as authentication for the `POST` request following in step 2.

#### Sample Code

```
Request
GET /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
    manage_software_components/0001 HTTP/1.1
Accept: application/json
X-Csrftoken: fetch
Host: host.sap
Response
HTTP/1.1 200 OK
X-csrf-token: xCsrfToken
    that was retrieved in the first request in the
    header parameters. The
    software component is passed in the path of the
    request URL. The branch you
    want to clone is passed in the body of the
    request. If you don't enter a
    branch name, the main branch is automatically
    selected. Alternatively, you
```

can clone a specific tag or commit ID by entering the tag name or commit ID. Make sure that the tag or commit ID exists on the branch. If that's not the case, no clone action will be executed.

## 2. Cloning a Software Component

To trigger the clone, you need to insert the `x-csrf-token`

### ↔ Sample Code

```
Request
POST /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
manage_software_components/0001/SoftwareComponents/%2FDMO%2FCOMPONENT/
SAP_self.clone HTTP/1.1
Accept: application/json
X-Csrf-Token: {{xcsrfToken}}
Host: host.abapcp.int.sap
Content-Length: 172
{
    "commit_id": "",
    "branch_name": "main",
    "tag_name": "",
    "username": "myUsername",
    "password": "myToken",
    "auth_method": "token"
}
Response
HTTP/1.1 200 OK
Content-Type: application/json
{
    "@odata.context": ".../$metadata#Actions/$entity",
    "@odata.metadataEtag": "W/\"20240111143934\"",
    "uuid": "96bbf3e1-5202-1ede-ad8e-b46db5437410",
    "sc_name": "/DMO/COMPONENT",
    "import_type": "Clone",
    "branch_name": "main",
    "namespace": "",
    "status": "Q",
    "status_descr": "Scheduled",
    "criticality": 0,
    "user_name": "WebAPI - CC0000000009",
    "commit_id": "",
    "tag_name": "",
    "relative_date_change_time": "",
    "change_time": "2024-01-16T13:46:26Z",
    "start_time": "2024-01-16T13:46:26Z",
    "execution_mode": "",
    "import_mode": "",
    "batch_id": null,
    "batch_sequence": 0,
    "SAP__Messages": []
}
```

- [2.1 Cloning Your Own Software Component from Git Repository Using Bring Your Own Git \(BYOG\)](#)

Determine whether you have created a Bring Your Own Git (BYOG) software component as described in the according documentation.

If the software component is to be cloned from an externally managed Git repository, the clone request must be adjusted. The request must be supplemented with the parameters `username`, `password` and `auth_method`. These parameters are used to authenticate with external Git providers.

Two authentication options, which are specified with the `auth_method` parameter, are supported.

Following, these two authentication options are listed.

auth_method	Meaning	Usage
basic	Authentication with username and password	<p>↔ Sample Code</p> <pre>{   "username":     "myUsername",    "password":     "myPassword",    "auth_method":     "basic" }</pre>
token	Authentication with username and token	<p>↔ Sample Code</p> <pre>{   "username":     "myUsername",    "password":     "myToken",    "auth_method":     "token" }</pre>

If auth\_method is not defined, the token method is automatically selected.

<p>↔ Sample Code</p> <pre>Request   POST /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/   manage_software_components/0001/SoftwareComponents/%2FDMO%2FCOMPONENT/   SAP__self.clone HTTP/1.1   Accept: application/json   X-Csrftoken: {{xcsrfToken}}   Host: host.abapcp.int.sap   Content-Length: 172   {     "commit_id": "",     "branch_name": "main",     "tag_name": "",     "username": "myUsername",     "password": "myToken",     "auth_method": "token"   } Response   HTTP/1.1 200 OK   Content-Type: application/json   {     "@odata.context": ".../\$metadata#Actions/\$entity",     "@odata.metadataEtag": "W/\"20240111143934\"",   }</pre>
--

```

    "uuid": "96bbf3e1-5202-1ede-ad8e-b46db5437410",
    "sc_name": "/DMO/COMPONENT",
    "import_type": "Clone",
    "branch_name": "main",
    "namespace": "",
    "status": "Q",
    "status_descr": "Scheduled",
    "criticality": 0,
    "user_name": "WebAPI - CC0000000009",
    "commit_id": "",
    "tag_name": "",
    "relative_date_change_time": "",
    "change_time": "2024-01-16T13:46:26Z",
    "start_time": "2024-01-16T13:46:26Z",
    "execution_mode": "",
    "import_mode": "",
    "batch_id": null,
    "batch_sequence": 0,
    "SAP__Messages": []
}

```

### 3. Tracking the Status of the Action

To track the status of the action, you can do a GET request using the UUID contained in the response.

#### ↔ Sample Code

```

Request
GET /sap/opu/odata4/sap/a4c_mswc_api/
srvd_a2x/sap/manage_software_components/0001/Actions/96bbf3e1-5202-1ede-
ad8e-b46db5437410 HTTP/1.1
Accept: application/json
Host: host.abapcp.int.sap

```

The response contains the same entity as the second request. The action is successful, when the *status* in the response body has the value *S*. The status description *status\_descr* will then return *Successful*. In case of an error *status* will have the value *E* and *status\_descr* the value *Error*.

## 6.3.1.23.3.2 Pulling Software Components to an ABAP Environment System

- You have an ABAP environment system.
- You've created a communication user as described in [How to Create Communication Users](#).
- You've created a communication system as described in [How to Create Communication Systems](#).
- You've created a communication arrangement as described in [How to Create a Communication Arrangement](#).
- You have selected the communication scenario SAP\_COM\_0948 for your communication arrangement and have mapped it to your communication system.
- You have cloned the software component to the ABAP Environment system. See Cloning Software Components to an ABAP Environment System.

You can use the communication scenario SAP\_COM\_0948 to pull software components to an ABAP environment system.

## 1. Authentication to the server

The first step serves the authentication on the server. The response header contains an `x-csrf-token`, which is used as authentication for the `POST` request following in step 2.

### ↔ Sample Code

```
Request
GET /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
manage_software_components/0001 HTTP/1.1
Accept: application/json
X-Csrftoken: fetch
Host: host.sap
Response
HTTP/1.1 200 OK
X-csrf-token: xCsrfToken
```

## 2. Pull a Software Component

To trigger the pull of a software component, you need to insert the `x-csrf-token` that was retrieved in the first request in the header parameters. The software component you want to pull is passed in path of the request URL.

### ↔ Sample Code

```
Request
POST /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
manage_software_components/0001/SoftwareComponents/%2FDMO%2FCOMPONENT%
SAP_self.pull HTTP/1.1
Accept: application/json
X-Csrftoken: {{xCsrfToken}}
Host: host.abapcp.int.sap
Content-Length: 39
{
  "commit_id": "",
  "tag_name": ""
}
Response
HTTP/1.1 200 OK
Content-Type: application/json
{
  "@odata.context": ".../$metadata#Actions/$entity",
  "@odata.metadataEtag": "W/\"20240111143934\"",
  "uuid": "763eca2a-340b-1eee-ad8e-c864d7a77c38",
  "sc_name": "/DMO/COMPONENT",
  "import_type": "Pull",
  "branch_name": "master",
  "namespace": "",
  "status": "Q",
  "status_descr": "Scheduled",
  "criticality": 0,
  "user_name": "WebAPI - CC0000000009",
  "commit_id": "",
  "tag_name": "",
  "relative_date_change_time": "",
  "change_time": "2024-01-16T13:50:54Z",
  "start_time": "2024-01-16T13:50:54Z",
  "execution_mode": "",
  "import_mode": "",
  "SAP_Messages": []
}
```

## 3. Tracking the Status of the Action

To track the status of the action, you can do a `GET` request using the `UUID` contained in the response.

#### ↔ Sample Code

```
Request
  GET /sap/opu/odata4/sap/a4c_mswc_api/
  srvd_a2x/sap/manage_software_components/0001/Actions/96bbf3e1-5202-1ede-
  ad8e-b46db5437410 HTTP/1.1
  Accept: application/json
  Host: host.abapcp.int.sap
```

The response contains the same entity as the second request. The action is successful, when the *status* in the response body has the value *S*. The status description *status\_descr* will then return *Successful*. In case of an error *status* will have the value *E* and *status\_descr* the value *Error*.

### 6.3.1.23.3.3 Working with Branches on a Software Component

- You have an ABAP environment system.
- You've created a communication user as described in [How to Create Communication Users](#).
- You've created a communication system as described in [How to Create Communication Systems](#).
- You've created a communication arrangement as described in [How to Create a Communication Arrangement](#).
- You have selected the communication scenario SAP\_COM\_0948 for your communication arrangement and have mapped it to your communication system.
- You have cloned the software component to the ABAP Environment system. See Cloning Software Components to an ABAP Environment System.

You can use the communication scenario SAP\_COM\_0948 to work with software components on an SAP BTP, ABAP environment system.

#### 1. *Authentification on the server*

The first step serves the authentication on the server. The response header contains an *x-csrf-token*, which is used as authentication for the POST request following in step 2 and 3.

#### ↔ Sample Code

```
Request
  GET /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
  manage_software_components/0001 HTTP/1.1
  Accept: application/json
  X-Csrf-Token: fetch
  Host: host.sap
Response
  HTTP/1.1 200 OK
  X-csrf-token: xCsrfToken
```

#### 2. *Create a Branch*

To create a branch, you perform a POST request on the Branches entity. In the header parameters, you should include the `x-csrf-token` that was retrieved in the first request. Fill in the following data in your request body:

1. `sc_name`: the name of your software component
2. `branch_name`: the name of your new branch
3. `derived_from`: the branch that your new branch should derive from

#### ↔ Sample Code

```
Request
POST /sap/opu/odata4/sap/a4c_mswc_api/srvd_a2x/sap/
manage_software_components/0001/Branches HTTP/1.1
Accept: application/json
X-Csrf-Token: {{xcsrfToken}}
Host: host.abapcp.int.sap
Content-Length: 103
{
    "sc_name" : "/DMO/COMPONENT",
    "branch_name" : "feature",
    "derived_from" : "main"
}
Response
HTTP/1.1 201 Created
Content-Type: application/json
{
    "@odata.context": "$metadata#Branches/$entity",
    "@odata.metadataEtag": "W/\"20240111143934\"",
    "sc_name": "/DMO/COMPONENT",
    "branch_name": "feature",
    "is_active": false,
    "criticality": 0,
    "delta": "Please checkout first",
    "delta_criticality": 0,
    "derived_from": "main",
    "name_of_new_branch": "",
    "commit_id": "",
    "commit_message": "",
    "last_commit_by": "",
    "relative_date_last_rem_commit": "",
    "last_commit_on": null,
    "commit_id_on_system": "-",
    "created_by": "CC0000000009",
    "created_on": "2024-01-16T14:00:00Z",
    "execution_mode": "",
    "import_mode": "",
    "SAP__Messages": []
}
```

#### 3. Check Out a Branch

Simply creating a branch did not change the state of the system. In order to work with the newly created branch, you need to import the branch to the system. This action is called *checkout branch*. You need to include the software component and the name of the branch in the path of the request URL.

#### ↔ Sample Code

```
Request
POST /sap/opu/odata4/sap/a4c_mswc_api/
srvd_a2x/sap/manage_software_components/0001/Branches/%2FDMO%2FCOMPONENT/
main/SAP__self.checkout_branch HTTP/1.1
Accept: application/json
X-Csrf-Token: {{xcsrfToken}}
Host: host.abapcp.int.sap
```

```

Content-Length: 44
{
  "import_mode" : "",
  "execution_mode": ""
}
Response
HTTP/1.1 200 OK
Content-Type: application/json
{
  "@odata.context": "...",
$metadata#com.sap.gateway.srvd_a2x.a4c_a2g_mswc_api.v0001.ActionsType",
  "@odata.metadataEtag": "W/\"20240111143934\"",
  "uuid": "96bbf3e1-5202-1ede-ad8f-001e560df4e6",
  "sc_name": "/DMO/FCOMPONENT",
  "import_type": "Checkout",
  "branch_name": "main",
  "namespace": "",
  "status": "Q",
  "status_descr": "Scheduled",
  "criticality": 0,
  "user_name": "WebAPI - CC0000000009",
  "commit_id": "",
  "tag_name": "",
  "relative_date_change_time": "",
  "change_time": "2024-01-16T14:03:22Z",
  "start_time": "2024-01-16T14:03:21Z",
  "execution_mode": "",
  "import_mode": "",
  "SAP__Messages": []
}

```

#### 4. Tracking the Status of the Action

To track the status of the action, you can do a GET request using the UUID contained in the response.

##### ↔ Sample Code

```

Request
GET /sap/opu/odata4/sap/a4c_mswc_api/
srvd_a2x/sap/manage_software_components/0001/Actions/96bbf3e1-5202-1ede-
ad8e-b46db5437410 HTTP/1.1
Accept: application/json
Host: host.abapcp.int.sap

```

The response contains the same entity as the second request. The action is successful, when the *status* in the response body has the value *S*. The status description *status\_descr* will then return *Successful*. In case of an error *status* will have the value *E* and *status\_descr* the value *Error*.

### 6.3.1.24 User Interface Monitoring

[Display Launchpad Content Exposure Logs \[page 2831\]](#)

### 6.3.1.24.1 Display Launchpad Content Exposure Logs

With this app, you can check the application log created when publishing launchpad content to SAP BTP. You can check, for example, when the publishing run started, if the publishing was successful or if there were any issues.

#### ⓘ Note

Log entries about apps that are not exposed and have a severity level of *Information* do not indicate an error. Applications may not be exposed for various reasons. For example, inactive applications may not be exposed in either version of exposure, or apps without a tile may not be displayed in exposure version 2.

### Implementation Information

To use the app, you need the following:

- Business catalog: SAP\_CORE\_BC\_UI\_MON
- Business role: SAP\_BR\_ADMINISTRATOR

To get the key information, including all the technical data you need for the installation and configuration, go to the [SAP Fiori apps reference library](#).

### Related Information

[Create Communication Arrangement](#)

### 6.3.1.25 Manage Security Audit Log

With this app, you can manage Security Audit Log related parameters.

### Key Features

You can use this app to:

- Each parameter can be active or inactive, so that changing the parameter value is possible without having the change immediately active
- Maintain parameter `DELETE`:
  - Maintain the default audit log retention time in days. After the number of days the Security Audit Log Events will be removed from database to reduce the consumption of the SAP HANA DB storage. The default retention period is 45 days. Confirm the default by activating the parameter `DELETE`, or define a custom retention period.

Mark the parameter you want to change and switch to *Edit* mode. Select the option for *Parameter Status* (active or inactive) and for *Parameter Value* (Keep, Replace, or Clear Value) and save your settings.

## Access Information

For information on what business role and business catalog need to be assigned to your user to access the app, see the overview table in the chapter [Applications for General Functions for Key Users](#).

## Supported Device Types

- Desktop
- Tablet
- Smartphone

## Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-SEC-SAL.

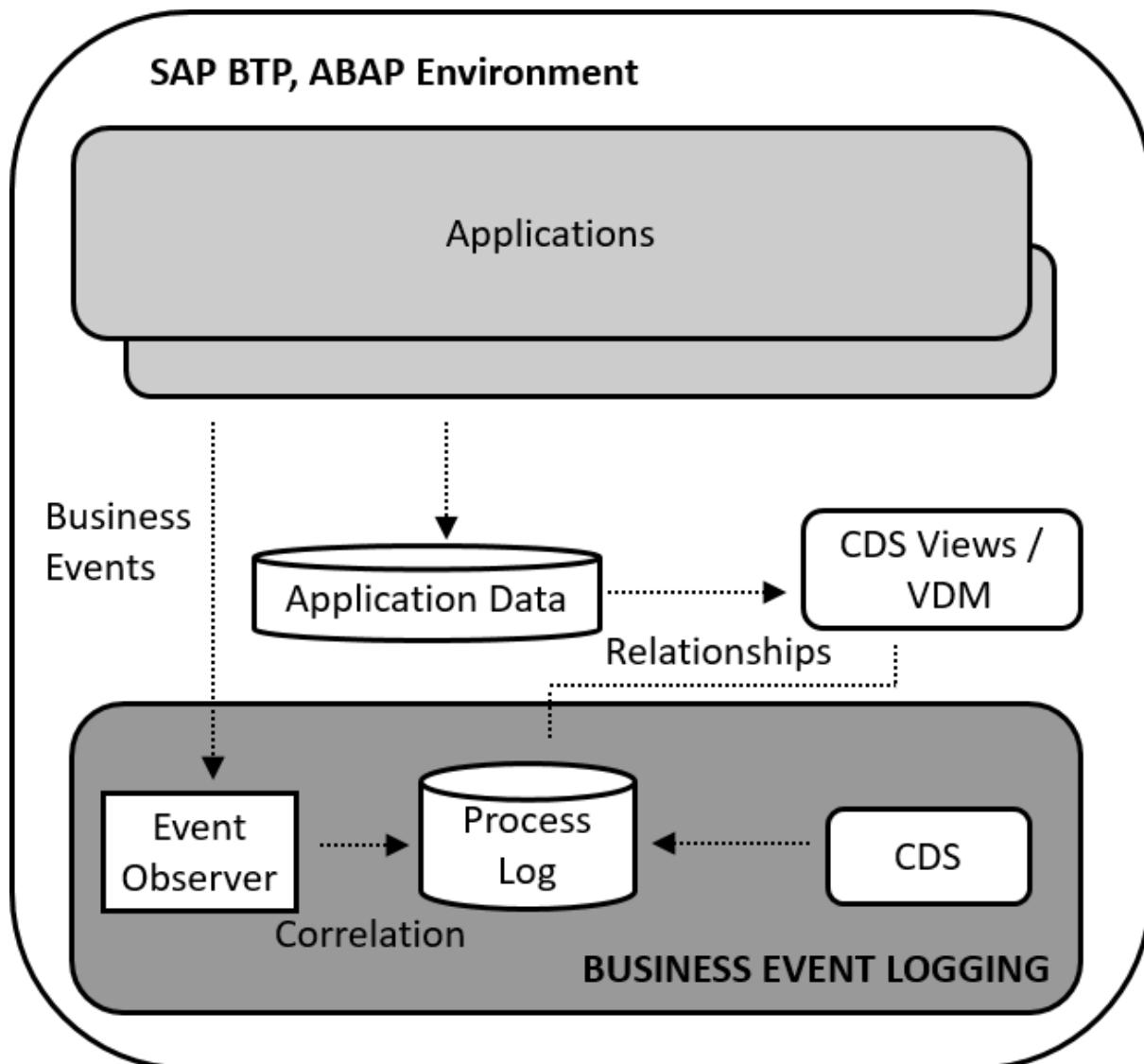
### 6.3.1.26 Business Event Logging

Business Event Logging enables you to log business events that are raised by SAP S/4HANA applications when business processes run in the local system. Business event is a message that is sent to notify a consumer that an object has changed. These events are listed in the SAP Business Accelerator Hub. Events are triggered during the execution of business activities. For example, you can find events such as **created**, **changed**, and **deleted** for Sales Order.

You can activate Business Event Logging using configuration activities for each object, such as Sales Order, Outbound Delivery, and so on. Business event is a message that is sent to notify a consumer that an object has changed.

The business event logs give you insights into the way processes run in the local system and how objects change. You can get an overview of all the logged business events, the number of events triggered, the types of business events triggered, and information on changes to fields

This graphic displays the components of Business Event Logging:



## Benefits

With Business Event Logging, you can collect records of these business events, irrespective of whether the consumer has subscribed to the event, to get insights into the way business processes run in the local system.. You can either store the default information or the complete business event data, depending on your configuration settings.

Essential properties of an event are stored by default. These properties are:

- Objects
- Object components
- Business object key

- Date of event
- Event type (corresponding to performed action)
- User who performed the action
- Relevant field changes
- Qualifiers (for generic events)

Field changes are also recorded as long as they are defined in the business events.

### 6.3.1.26.1 Business Event Logging Configuration

#### Prerequisites

To configure business event logging, create a business role and assign the catalog SAP\_BCR\_CA\_IC\_LND\_BEL\_BCO\_PC to the role. For instructions, refer to [How to Create a New Business Role \[page 2667\]](#). You can also create a business role using the template SAP\_BR\_BPC\_EXPERT. For instructions, refer to [How to Create a Business Role from a Template \[page 2668\]](#).

#### Configure the Application

##### Activate Business Event Logging

To activate logging for your objects:

1. Open the *Custom Business Configurations* app. You'll see a list of all business configurations that can be adapted.
2. Search for *Activate Business Event Logging* using the search bar.
3. Select the custom business configuration from the list and choose *Edit*.
4. To activate logging for an object, select the *Activate Logging* checkbox and choose *Save*.
5. (Optional) To enable logging for all business event data, select the *Log All Data* checkbox for the object.

##### ⚠ Caution

When you enable logging for all business event data, it consumes additional storage.

6. To create a new entry, choose *New Entries*.
7. Enter the *Object* and select the *Activate Logging* checkbox.
8. Choose *Save*.

For documentation about this item, refer to [Show Documentation](#).

## 6.3.1.26.2 Business Event Logging Applications

With these apps, you can view business event logs and business events by SAP object nodes in your local system.

### Key Features

Business Event Logging contains these features:

- Displaying Business Event Logs
- Displaying Business Events by Objects
- Display Changes to Objects

### Components

All users can use the following applications in the ABAP environment for Business Event Logging:

- `Display Business Event Logs` app to view all the events logged in the SAP S/4HANA system.
- `Display Business Events by Objects` app to view all the sequences of business events in the context of the instances of object components.
- `Display Changes to Objects` app to view all changes for an Object.

### Prerequisites

To access applications, create a business role and assign business catalog **SAP\_CA\_BC\_BEL\_PC** to the role. For instructions, refer [here](#).

To create a business role using a template, select the **SAP\_BR\_BUSINESS\_PROCESS\_SPEC** template. For instructions, refer [here](#).

### 6.3.1.26.2.1 Display Business Event Logs

With this app, you can display logged business events. The events can be filtered by business event type for each SAP object type.

### Key Features

- a) You can use this app to:

- Get an overview of logged business events
- View details of logged business events
- Go to business event sequence for a node

This app uses the **C\_BusEventLogEventAnlys** CDS view.

## Supported Device Types

- Desktop
- Tablet
- Smartphone

### 6.3.1.26.2.2 Display Business Events by SAP Object

With this app, you can display business events grouped by object instances. You can filter the results by processing variants. The detailed view provides information about the sequences of business events for instances of object components.

## Key Features

a) You can use this app to view:

- Instances of object components containing business events
- Sequence of business events logged for an instance of object component
- Processing variants, number of events for each instance of object component, and the processing cycle time

This app uses the **C\_BusEvtLogObjectNodeAnlys** CDS view.

## Supported Device Types

- Desktop
- Tablet
- Smartphone

### **6.3.1.26.2.3 Display Changes to Objects**

**App ID:** F8211

With this app, you can view the recorded field changes for an object.

#### **Key Features**

You can use this app to view:

- The field changes that have recorded for an object
- Old and new values for a changed field

This app uses the `C_BuEvLgSAPObjNdePrptyChg` CDS view.

#### **Supported Device Types**

- Desktop
- Tablet
- Smartphone

### **6.3.1.26.3 Monitoring for Business Event Logging**

You can monitor the problems that occur in business event logging using:

- Message Monitoring: You can use Message Monitoring Overview app to view runtime errors, to monitor processing of events, and to view missing events.
- Message Monitoring: You can use Message Monitoring Overview app to view runtime errors, to monitor processing of events, and to view missing events.

#### **6.3.1.26.3.1 Message Monitoring**

You monitor the problems that occur in business event logging by using Application Interface Framework (AIF) . You can check the error status to identify failures. If there are failures in the queue, the business events aren't logged. However, the events aren't lost.

## Prerequisites

To access Message Monitoring, create a business role and assign the role to a catalog. For more information about catalogs and other settings, refer to [Message Monitoring \[page 2708\]](#). Use the BEL\_IBD\_QUEUE\_MONITOR recipient to view Business Event Logging messages.

## Message Monitoring

1. Log on to SAP Fiori launchpad.
2. Open the [Message Monitoring Overview](#) app. For more information, refer to Message Monitoring Overview.
3. Select *From* (Date/Time) and *To* (Date/Time) values to limit the time range for the monitoring records.

### ⓘ Note

The default time range is "last 2 weeks". If you want a custom time range, you can also select the values in the *From* and *To* fields.

4. Choose *Go* to view the messages in this time range.
5. Select the following cards to monitor the processing of business events for the business event log:
  - Namespace: /BEL
  - Interface:
    - BEL\_IBD
    - BEL\_LG\_MGR
- If you want to view only the error or success messages, choose [Error](#) or [Success](#).
6. On the [Message Monitoring](#) screen, in the [Log Messages](#) column, you can see all business event logging messages for your selected time range. All types of messages, such as success, error, warning, canceled, and in process are included.
7. You can filter by [Status](#).
8. Choose [Message Details](#) to see all the message logs.

If a queue fails, the user can restart the queue manually with the AIF user interface. If errors occur during restart, create an incident under component **CA-GTF-BEL**.

## 6.3.1.26.3.2 Application Log

You can use the [Application Logs](#) app to display application logs for Business Event Logging. If you need to see at a glance errors or information that occurred during Business Event Logging execution, this app gives you a structured overview. To go to the [Application Logs](#) app, select the corresponding icon in the Log on the initial screen of the [Application Jobs](#) app. For more information, see [Application Logs \[page 1536\]](#). To display the logs specific to Business Event Logging, filter based on the BEL\_LOG category.

Log Object	Log Subobject	Description
BEL_LOG	BUFFER	Displays information about metadata buffer updates.
BEL_LOG	DELETE	Displays when and what business event log entries have been deleted.
BEL_LOG	ERROR	Displays information about errors occurring during business event logging.

## 6.3.1.26.4 CDS Views for Business Event Logging

### 6.3.1.26.4.1 Business Event Header Data (v2)

CDS View Name	C_BUSEVTLOGEVENTDEX_2
Analytical Data Category	@Analytics.dataCategory: #FACT
Represented Objects	<p>This view represents the following SAP object type:</p> <ul style="list-style-type: none"> <li>• BusEvtLogEvent (TechnicalObject)</li> </ul>

## Purpose

This CDS view is used to access business event header data via key user extensibility.

This CDS view provides the data to answer the following business questions:

- Which business event has been logged?

To help you decide which CDS view to use for your purposes, SAP has introduced the annotation `ObjectModel.supportedCapabilities` that indicates the most appropriate use cases for each CDS view. To find out what use cases are best supported by this CDS view, access the entry of the CDS view in the [View Browser](#) app and find the values for this annotation under the [Annotation](#) tab.

## Structure

### Important Fields

Important fields in this view include the following:

Field Name	Description
EventOperation	Event operation, for example: Created, Deleted, Changed, and so on.
SAPObjectType	The object type, a business object, or a technical object corresponding to the specified SAP object node type, for example: Outbound Delivery for Outbound Delivery Item.
SAPObjectNodeType	Node type of an object type, for example, for an Outbound Delivery SAP object type, the SAP object node types are Outbound Delivery (header) and Outbound Delivery Item (item).
SAPBusinessObjectNodeKey1	First part of the key of an object node instance, for example, in one event for an Outbound Delivery Item, the first part of the key is 80013910 (outbound delivery number).
SAPBusinessObjectNodeKey2	Second part of the key of an object node instance, for example, in one event for an Outbound Delivery Item, the second part of the key is 00020 (item number). If an event refers to an SAP object node type with only one key part (such as: Outbound Delivery [header]), then all subsequent key parts are empty.
<div style="background-color: #f0f0f0; padding: 10px;"><p><b> ⓘ Note</b></p><p>There can be many more node keys, depending on how many parts the object key has. For example: SAPBusinessObjectNodeKey3, SAPBusinessObjectNodeKey4, and so on.</p><p>SAPObjectType and SAPBusinessObjectNodeKey (1-7) identify a node instance of an SAP object. Outbound Delivery Item is an example of an SAP object node.</p></div>	
BusEvtLogCreationDateTime	Event execution timestamp
CreatedByUser	Users who created the event
EventProducerNamespace	Provides the event's namespace. This can either be a Standard or Custom

## Examples

Example of events include:

BusinessPartner

- Business partner created
- Business partner changed

For more information on other events, refer to SAP Business Accelerator Hub.

### 6.3.1.26.4.2 Business Event Property Changes (v2)

CDS View Name	C_BUSEVTLOGPAYLOADDEX_2
Analytical Data Category	@Analytics.dataCategory: #FACT
Represented Objects	<p>This view represents the following SAP object type:</p> <ul style="list-style-type: none"> <li>• BusEvtLogEvent (TechnicalObject)</li> </ul>

#### Purpose

This CDS view provides a subset of event payload details.

This CDS view provides the data to answer the following business questions:

- What are the event qualifiers?
- What are the field changes?

To help you decide which CDS view to use for your purposes, SAP has introduced the annotation `ObjectModel.supportedCapabilities` that indicates the most appropriate use cases for each CDS view. To find out what use cases are best supported by this CDS view, access the entry of the CDS view in the [View Browser](#) app and find the values for this annotation under the [Annotation](#) tab.

#### Structure

##### Important Fields

Important fields in this view include the following:

Field Name	Description
BusinessEventUUID	Unique identifier for business events.
SAPObjectType	The object type, a business object, or a technical object corresponding to the specified SAP object node type. For example: Outbound Delivery for Outbound Delivery Item.
BusEvtLogGlobalFieldName	Field of an SAP object. Example: Amount

Field Name	Description
BusEvtLogFieldHasOldValue	This is a Boolean flag that indicates whether there was a value for the field.
BusEvtLogFieldIsQualifier	This is a Boolean flag that indicates whether the field is an event qualifier. A qualifier is an annotated field in a generic event that defines the event in more detail. The value of the qualifier field describes a generic event in detail.  Example: For the event Outb Deliv Goods Issue Status Changed the qualified field or Qualifier is OverallGoodsMovementStatus, with the values: Not yet started, Partially Completed, Completed, or Not Relevant
BusEvtLogLastChangedDateTime	The date and timestamp of when the payload entry was last updated.

### 6.3.1.26.4.3 Business Event Data

CDS View Name	C_BuEvLgEvtFullPayloadJSONDEX
Analytical Data Category	@Analytics.dataCategory: #FACT
Represented Objects	<p>This view represents the following SAP object type:</p> <ul style="list-style-type: none"> <li>• BusEvtLogEvent (TechnicalObject)</li> </ul>

## Purpose

This CDS view provides full business event data (payload) in JSON format.

This CDS view provides the data to answer the following business questions:

- What is the full business data contained in the event?

To help you decide which CDS view to use for your purposes, SAP has introduced the annotation `ObjectModel.supportedCapabilities` that indicates the most appropriate use cases for each CDS view. To find out what use cases are best supported by this CDS view, access the entry of the CDS view in the [View Browser](#) app and find the values for this annotation under the [Annotation](#) tab.

## Structure

### Important Fields

Important fields in this view include the following:

Field Name	Description
SAPObjectType	The object type, a business object, or a technical object corresponding to the specified SAP object node type. For example: Outbound Delivery for Outbound Delivery Item.
BusEvtLogPayloadJSONString	Event Payload details in JSON format.
BusEvtLogLastChangedDateTime	Timestamp of when the event was changed.

## 6.3.2 Communication Operations

### [Communication Management \[page 2844\]](#)

Learn more about the basic principles of communication management when integrating your system or solution with other systems to enable data exchange in your ABAP environment.

### [Integration in BTP Apps via SAP Destination Service \[page 2847\]](#)

In this topic you learn how to configure destinations in the SAP Destination service that shall be used by applications to consume services provided by the ABAP environment, and how to maintain corresponding communication arrangements in the ABAP environment. For instance, how to configure a destination that is used by an SAP Fiori application from the HTML5 repository that accesses an OData service of the ABAP environment.

### [Overview of Communication Scenarios Managed by SAP \[page 2862\]](#)

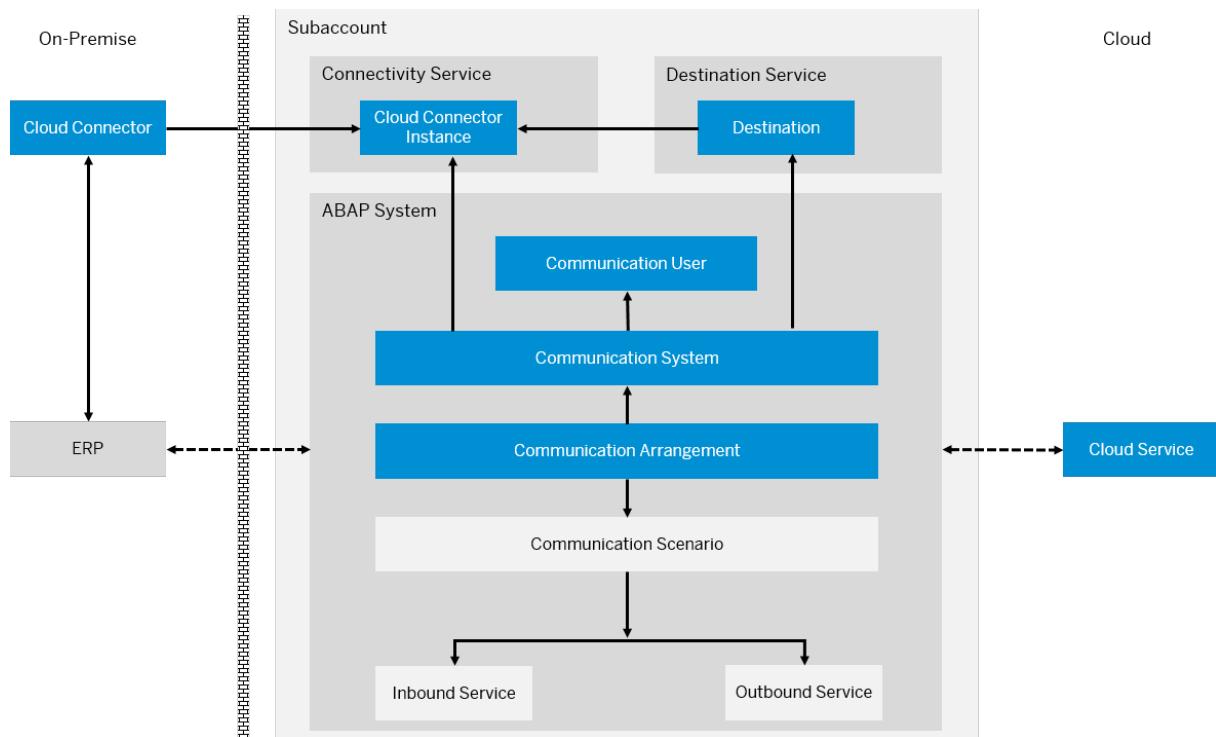
Find a quick overview of all the communication scenarios in the ABAP environment.

### [Integration Scenarios \[page 2863\]](#)

You can integrate the ABAP environment with the following services.

## 6.3.2.1 Communication Management

Learn more about the basic principles of communication management when integrating your system or solution with other systems to enable data exchange in your ABAP environment.



### 6.3.2.1.1 Communication Scenario

A communication scenario, which is created in the development system using ABAP Development Tools and transported to other systems, is a design time description of how two communication partners communicate with each other. It consists of inbound and/or outbound services as well as supported authentication methods.

It provides technical information, such as the used inbound and outbound services and their service type, for example OData or SOAP, and the number of allowed communication arrangement instances. If the scenario exposes inbound services, it specifies the authorizations that are required to run the services.

The following types of communication scenarios are available:

- **Managed by SAP**, where SAP provides a ready-to-use communication scenario and you create and maintain a communication arrangement.
- **Managed by the customer**, where you develop a communication scenario and create and maintain a communication arrangement.

## Related Information

[Display Communication Scenarios \[page 2595\]](#)

[Overview of Communication Scenarios Managed by SAP](#)

[Developing APIs for Inbound Communication \[page 1034\]](#)

[Outbound Communication](#)

### 6.3.2.1.2 Communication System

A communication system is a specification of a system that represents a communication partner and the technical information required for the communication, such as the host name, port, users for inbound or outbound communication, certificates, etc.

If the communication system represents an on-premise system that is protected by a firewall, the system can be connected by assigning a cloud connector.

Instead of maintaining the credentials for outbound communication in the communication system itself, the communication system can also refer to a destination from the destination service. This can be a destination on subaccount level, or a destination from a destination service instance.

An administrator user in the ABAP environment has to create the communication system in the [Communication Systems \[page 2598\]](#) app in SAP Fiori launchpad. Note that the communication system is not transported between ABAP systems but created locally. The communication partner can vary for each system.

### 6.3.2.1.3 Communication User

A communication user is a specific type of technical user that is assigned to a communication system. The user can be assigned a password or X.509 certificate.

A communication user is added as a user for inbound communication to communication systems.

An administrator user in the ABAP environment creates the communication user in the [Maintain Communication Users \[page 2593\]](#) app in SAP Fiori launchpad. Note that the communication user is not transported between systems but created locally. The technical users and their credentials can vary for each system.

### 6.3.2.1.4 Communication Arrangement

A communication arrangement is a runtime description of a specific communication scenario. It describes which communication partners communicate with each other in the scenario, and how they communicate.

To describe this runtime behavior, you have to create an arrangement for a scenario, assign the communication system and communication users, and select the authentication method that shall be used.

If the communication scenario exposes inbound services, the communication user is granted the authorizations that have been specified for the communication scenario.

An administrator user in the ABAP environment creates the communication arrangement in the [Communication Arrangements \[page 2595\]](#) app in SAP Fiori launchpad. Note that it is not transported between systems but created locally. For certain SAP-managed scenarios that integrate an SAP BTP service, you can also create a communication arrangement from a service key of the service instance that you would like to connect.

For inbound-only scenarios, you can create a communication arrangement by creating a corresponding service key for the ABAP environment service instance.

## Related Information

[Maintain a Communication Arrangement for Inbound Communication](#) 

### 6.3.2.1.5 Destination Service

Using the SAP destination service, you can retrieve and store technical information about the target resource (destination) that you want to connect with your application to a remote service or a system.

The destination service allows you to read and manage the address of a remote service and the user authentication information for the connection on subaccount and service instance level.

## Related Information

[Destination Service](#)

[Create a Destination \[page 891\]](#)

### 6.3.2.1.6 Destination

A destination is stored in the SAP destinations service and contains the connection details for the communication partner.

You can use a destination to:

- Connect your application to the Internet (via HTTP, RFC, or WebSocket RFC) or to an on-premise system (via HTTP or RFC)
- Send and retrieve emails by configuring a mail destination

#### Note

You can create a destination for a destination service on instance level or subaccount level.

A destination can be referenced in a communication system. To refer a destination from a destination service instance, a communication arrangement for SAP-managed scenario SAP\_COM\_0276 needs to be maintained to enable communication with the destination service instance. See [Create a Destination \[page 891\]](#). In this case, the destination name and the service instance name need to be maintained in the communication system.

## Related Information

[Managing Destinations](#)

### 6.3.2.1.7 Cloud Connector

The Cloud Connector serves as a link between cloud applications and on-premise systems.

In the Cloud Connector, the subaccount of the ABAP system is connected so that the Cloud Connector can be used within a communication system or destination in the subaccount.

It provides an easy setup with a clear configuration of the systems that are exposed to SAP BTP.

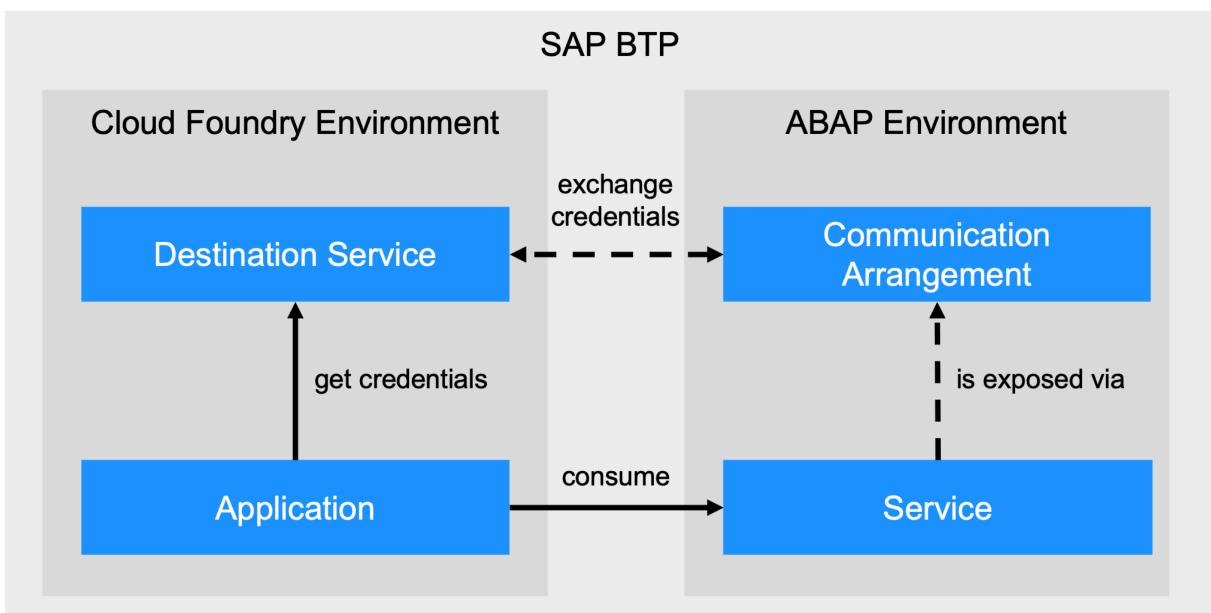
## Related Information

[Cloud Connector](#)

[Integrating On-Premise Systems \[page 1184\]](#)

### 6.3.2.2 Integration in BTP Apps via SAP Destination Service

In this topic you learn how to configure destinations in the SAP Destination service that shall be used by applications to consume services provided by the ABAP environment, and how to maintain corresponding communication arrangements in the ABAP environment. For instance, how to configure a destination that is used by an SAP Fiori application from the HTML5 repository that accesses an OData service of the ABAP environment.



## Choosing the Right Authentication Method

Before you start with the configuration, you need to decide on the right authentication method first. The right authentication method can be decided by answering the following questions:

### What Authentication Methods Are Supported Technically?

The authentication method needs to be supported by both parties. The following authentication methods are commonly supported by the SAP Destination service and the ABAP environment. For more information, see [Supported Protocols and Authentication Methods \[page 886\]](#).

- Authentication via communication user
  - Basic authentication (user name & password)
  - Client certificate authentication (X.509 certificate)
- Authentication via business user (principal propagation)
  - SAML Assertion authentication
  - OAuth 2.0 SAML Bearer Assertion grant
    - Client secret authentication (client ID & secret)
    - Mutual TLS (mTLS) client authentication (client ID & X.509 certificate)

The following applies to each authentication method:

- Basic authentication: A user name and password are sent to the ABAP environment for authentication.
- Client certificate authentication: An X.509 certificate is used.
- SAML assertion authentication: A SAML bearer assertion is sent.

In all three cases the credentials are directly sent to the ABAP environment. However, when using OAuth 2.0, then two endpoints are involved: the authorization server and the resource server. First, the SAP Destination service calls the token endpoint of the authorization server and provides an authorization grant to obtain an access token. To authenticate at the authorization server, a client secret or an X.509 certificate can be used.

The supported authorization grant is SAML bearer assertion. Second, the client application uses the access token to authenticate at the ABAP environment to access the actual resource.

## Who Shall Be Acting?

Decide if the actions in the ABAP environment shall be executed under a communication user independent from the user of the client application, or if the actions shall be performed under an individual business user corresponding to the user of the client application (principal propagation).

If the service shall be processed as communication user, then choose one of the following authentication methods:

- Basic authentication
- Client certificate authentication

If the principal shall be propagated to the target, then use one of the following authentication methods:

- SAML assertion authentication
- OAuth 2.0 SAML Bearer Assertion

Although OAuth SAML 2.0 Bearer Assertion is a well-known standard and enables additional access restriction using OAuth scopes, it also has some drawbacks compared to the proprietary SAML assertion authentication method:

- It requires a corresponding communication scenario and arrangement to provide the scope to the OAuth client.
- Only OData services are supported.
- It requires an additional round trip to exchange the authorization grant against an access token.
- The configuration is much more complicated and, thus, more error prone.

If you don't have any special requirements to use the OAuth standard, then the usage of SAML assertion authentication is preferred over OAuth 2.0 SAML Bearer Assertion.

## How Strong Are the Security Requirements?

Certificate-based authentication is more secure than using a password-based authentication method: a password is a shared secret and is known by both communication partners. Certificate-based client authentication uses public key infrastructure. The private key, which is owned by the client, isn't shared with the communication partner. Moreover, a password change between the communication partners is disruptive, while a non-disruptive change can be achieved with certificate-based authentication since more than one client certificate can be assigned to a communication user.

Hence, client certificate authentication is preferred over Basic Authentication and OAuth 2.0 SAML Bearer Assertion Grant with mTLS-based OAuth client authentication is preferred over OAuth 2.0 SAML Bearer Assertion Grant with client secret authentication.

The client certificates can be provided by the application, can be uploaded to the SAP Destination service, or generated by the SAP Destination service. The ABAP environment needs to trust the certificate authority (CA) of the client certificate. The list of trusted root certificate authorities can be found in [SAP Note 2801396](#).

## Configure the Destination and the Communication Arrangement

Once you have chosen the appropriate authentication method, you can continue to configure the destination, communication user, communication system, and communication arrangement accordingly:

- Inbound communication via communication user
  - [Configuring Basic Authentication \[page 2850\]](#)
  - Configuring client certificate authentication (X.509 certificate)
- Inbound communication via business user
  - [Configuring SAML Assertion Authentication \[page 2855\]](#)
  - [Configure OAuth 2.0 SAML Bearer Assertion Grant \[page 2858\]](#)
    - Client secret authentication (client ID & secret)
    - Mutual TLS (mTLS) client authentication (client ID & X.509 certificate)

### 6.3.2.2.1 Inbound Communication via Communication User

#### [Configuring Basic Authentication \[page 2850\]](#)

When using the basic authentication method, the client application provides the name of a communication user and its password to authenticate against the ABAP environment.

#### [Configuring Client Certificate Authentication \[page 2852\]](#)

When using the client certificate authentication method, the client application uses a client certificate to authenticate against the ABAP environment as corresponding communication user.

#### 6.3.2.2.1.1 Configuring Basic Authentication

When using the basic authentication method, the client application provides the name of a communication user and its password to authenticate against the ABAP environment.

#### Procedure

1. Create a communication user using the Maintain Communication Users app in your ABAP environment system. Enter the following data:

Field	Input
<i>User Name</i>	Provide a user name, for example, <b>MY_BASIC_AUTH_USER</b> .
<i>Description</i>	Provide a description.

Field	Input
<i>Password</i>	Provide a password or generate one.

2. Create a communication system.

Create a communication system using the Communication Systems app in your ABAP environment system. Enter the following data:

General Data

Field	Input
<i>System ID</i>	Provide a system ID, for example, <b>MY_COMMUNICATION_PARTNER</b> .
<i>System Name</i>	Provide a system name.

Technical Data

Field	Input
<i>General: Inbound Only</i>	Activate

Users for Inbound Communication

Authentication Method	User Name/Client ID
<i>User ID and Password</i>	User name of the communication user, for example, <b>MY_BASIC_AUTH_USER</b> .

3. Create a communication arrangement.

Create a communication arrangement using the Communication Arrangements app in your ABAP environment system. Enter the following data:

Field	Input
<i>Scenario ID</i>	Select the scenario, for example, <b>Z_MY_SCENARIO</b> .
<i>Arrangement Name</i>	Provide a name, for example, <b>MY_ARRANGEMENT</b> .
<i>Communication System</i>	Select the communication system, for example, <b>MY_COMMUNICATION_PARTNER</b> .
<i>Inbound Communication: User Name</i>	Select the communication user, for example, <b>MY_BASIC_AUTH_USER</b> .
<i>Inbound Communication: Authentication Method</i>	<b>User ID and Password</b>

4. Get technical data of the own communication system.

Open the own communication system in the Communication Systems app in your ABAP environment system. To open your own communication system choose [Own SAP Cloud System](#).

Denote the host name, for example 1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com.

5. Create a destination.

Create a destination using the Destinations editor in the SAP BTP cockpit. For more information, see [Using the Destination Editor in the Cockpit](#). Provide the following data:

Field	Input
Name	Enter the name of the destination that is used by the application, for example, <b>my-basic-destination</b> .
Type	<b>HTTP</b>
URL	Enter <b>https://&lt;hostname of the own communication system&gt;</b> , for example, <b>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com</b> .
Proxy Type	<b>Internet</b>
Authentication	<b>BasicAuthentication</b>
User	Enter the name of the communication user.
Password	Enter the password of the communication user.

## Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

[Communication Arrangements \[page 2595\]](#)

### 6.3.2.2.1.2 Configuring Client Certificate Authentication

When using the client certificate authentication method, the client application uses a client certificate to authenticate against the ABAP environment as corresponding communication user.

## Context

The client certificate can be provided by the application itself or by the SAP Destination service. The ABAP environment needs to trust the signing root certificate authority (CA). The list of trusted root certificate authorities can be found in [SAP Note 2801396](#).

## Procedure

1. Maintain the key store.

If the client application does not provide its own client certificate, then either upload a key store including the client certificate, the private key, and the certificate chain to the SAP Destination service, or generate one, for instance `my-keystore.p12`. For more information, see [Use Destination Certificates](#).

2. Get technical data of the own communication system.

Open the own communication system in the Communication Systems app in your ABAP environment system. To open your own communication system choose [Own SAP Cloud System](#).

Denote the host name, for example, `1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com`.

3. Create a destination.

Create a destination using the Destinations editor in the SAP BTP cockpit. For more information, see [Using the Destination Editor in the Cockpit](#). Provide the following data:

Field	Input
<code>Name</code>	Enter the name of the destination that is used by the application, for example, <code>my-client-certificate-destination</code> .
<code>Type</code>	<code>HTTP</code>
<code>URL</code>	Enter <code>https://&lt;hostname of the own communication system&gt;</code> , for example, <code>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com</code> .
<code>Proxy Type</code>	<code>Internet</code>
<code>Authentication</code>	<code>ClientCertificateAuthentication</code>
<code>Use client provided certificate</code>	Activate, if the client provides the client certificate. Otherwise, deactivate.
<code>Key Store Location</code>	If the certificate is provided by the SAP Destinations service and not the client itself, then select the name of the corresponding key store, for instance <code>my-keystore.p12</code> .
<code>Key Store Password</code>	If the certificate is provided by the SAP Destinations service and not the client itself, then provide the password of the key store.

4. Create a communication user.

Create a communication user using the Maintain Communication Users app in your ABAP environment system. Enter the following data:

Field	Input
User Name	Provide a user name, for example, <b>MY_CERT_USER</b> .
Description	Provide a description.
Certificate	Upload the client certificate to the communication user. During runtime, the service will be processed by the communication user that is assigned to the certificate.

5. Create a communication system.

Create a communication system using the Communication Systems app in your ABAP environment system. Enter the following data:

General Data

Field	Input
System ID	Provide a system ID, for example, <b>MY_COMMUNICATION_PARTNER</b> .
System Name	Provide a system name.

Technical Data

Field	Input
General: Inbound Only	Activate

Users for Inbound Communication

Authentication Method	User Name/Client ID
SSL Client Certificate	User name of the communication user, e.g. <b>MY_CERT_USER</b>

6. Create a communication arrangement.

Create a communication arrangement using the Communication Arrangements app in your ABAP environment system. Enter the following data:

Field	Input
Scenario ID	Select the scenario, for example, <b>Z_MY_SCENARIO</b> .
Arrangement Name	Provide a name, for example, <b>MY_ARRANGEMENT</b> .
Communication System	Select the communication system, for example, <b>MY_COMMUNICATION_PARTNER</b> .
Inbound Communication: User Name	Select the communication user, for example, <b>MY_CERT_USER</b> .
Inbound Communication: Authentication Method	<b>SSL Client Certificate</b>

## Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

[Communication Arrangements \[page 2595\]](#)

## 6.3.2.2.2 Inbound Communication via Business User (Principal Propagation)

[Configuring SAML Assertion Authentication \[page 2855\]](#)

When using the SAML assertion authentication method, the client application sends a signed SAML bearer assertion containing information about the business user to authenticate against the ABAP environment.

[Configure OAuth 2.0 SAML Bearer Assertion Grant \[page 2858\]](#)

When using the OAuth 2.0 SAML Bearer Assertion Grant authentication method, the SAP Destination service sends a signed SAML Bearer Assertion containing information about the business user to the token endpoint of the ABAP environment to obtain an access token.

### 6.3.2.2.2.1 Configuring SAML Assertion Authentication

When using the SAML assertion authentication method, the client application sends a signed SAML bearer assertion containing information about the business user to authenticate against the ABAP environment.

#### Context

You can provide your own certificate to sign the SAML assertion or use the standard subaccount-wide signing certificate. The ABAP environment needs to trust the issuer of the SAML bearer assertion.

#### Procedure

1. Maintain the key store for the signing certificate.

If the SAML assertion shall not be signed by the standard subaccount-wide signing certificate, then either upload a key store including the signing certificate, the private key, and the certificate chain, or generate one, for instance `my-signing-keystore.p12`. For more information, see [Use Destination Certificates](#).

2. Download the signing certificate.

If you use the standard subaccount-wide signing key, then download the signing certificate by choosing [Download Trust](#) in the SAP Destination editor.

3. Get data of the own communication system.

Open the own communication system in the Communication Systems app in your ABAP environment system. To open your own communication system, choose [Own SAP Cloud System](#).

Denote the following values:

- Host Name, for example, 1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com
  - SAML2 Audience, for example, <https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap-web.eu10.hana.ondemand.com>
4. Create a destination.

Create a destination using the Destinations editor in the SAP BTP cockpit. For more information, see [Using the Destination Editor in the Cockpit](#). Provide the following data:

Field	Input
Name	Enter the name of the destination, for example, <b>my-SAML-assertion-destination</b> .
Type	<b>HTTP</b>
URL	Enter <b>https://&lt;hostname of the own communication system&gt;</b> , for example, <b>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com</b> .
Proxy Type	<b>Internet</b>
Authentication	<b>SAMLAssertion</b>
Key Store Location	If you don't use the standard signing certificate, then select the name of the corresponding key store, for instance <b>my-signing-keystore.p12</b> .
Key Store Password	If you don't use the standard signing certificate, provide the password for the keystore.
Audience	Enter the OAuth 2.0 SAML2 Audience from the own communication system, for example, <b>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap-web.eu10.hana.ondemand.com</b> .
AuthnContextClassRef	<b>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</b>

Enter the following additional properties:

Additional Properties

Property	Input
<b>nameIdFormat</b>	Enter <b>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</b> if the e-mail is propagated or <b>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</b> if the user name is propagated. For more information, see <a href="#">User Propagation via SAML 2.0 Bearer Assertion Flow</a> .

5. Create a communication user.

For this authentication method, a communication user isn't needed since it is solely based on the trust relationship to the issuer of the SAML assertion – which is configured in the communication system.

6. Create a communication system.

Create a communication system using the Communication Systems app in your ABAP environment system. Enter the following data:

General Data

Field	Input
<i>System ID</i>	Provide a system ID, for example, <b>MY_COMMUNICATION_PARTNER</b> .
<i>System Name</i>	Provide a system name.

Technical Data

Field	Input
<i>General: Inbound Only</i>	Activate

Identity Provider

Field	Input
<i>SAML Bearer Assertion Provider</i>	Activate
<i>User ID Mapping Mode</i>	Select <i>User Name</i> .  If the <b>NameIdFormat</b> is unspecified, then the value of the <b>NameID</b> element in the SAML assertions' subject is mapped to the user name.
<i>SAML Bearer Issuer</i>	Enter the SAML entity ID of the SAML bearer issuer, which corresponds to the common name (the string after <b>CN=</b> ) of the signing certificate subject.
<i>Upload Signing Certificate</i>	Upload the signing certificate.

7. Create a communication arrangement.

For this authentication method, a communication arrangement isn't needed since it's solely based on the trust relationship to the issuer of the SAML assertion – which is configured in the communication system.

## Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

[Communication Arrangements \[page 2595\]](#)

## 6.3.2.2.2 Configure OAuth 2.0 SAML Bearer Assertion Grant

When using the OAuth 2.0 SAML Bearer Assertion Grant authentication method, the SAP Destination service sends a signed SAML Bearer Assertion containing information about the business user to the token endpoint of the ABAP environment to obtain an access token.

### Context

You can provide your own certificate to sign the SAML assertion or use the standard subaccount-wide signing certificate. The ABAP environment needs to trust the issuer of the SAML Bearer Assertion.

To authenticate the OAuth client as communication user at the token endpoint, a client secret (password of the communication user) or client certificate can be used. In the latter case, the client certificate needs to be provided by the SAP Destination service and the ABAP environment needs to trust the signing root certificate authority (CA). The list of trusted root certificate authorities can be found in [SAP Note 2801396](#).

Finally, the client application uses the access token to authenticate as a business user at the ABAP environment to access the actual resource.

### Procedure

1. Maintain the key store for the signing certificate.

If the SAML assertion shall not be signed by the standard subaccount-wide signing key, then either upload a key store including the signing certificate, the private key, or generate one, for instance **my-signing-keystore.p12**. For more information, see [Use Destination Certificates](#).

2. Download the signing certificate.

If you use the standard subaccount-wide signing key, then download the signing certificate by choosing [Download Trust](#) in the SAP Destination service.

3. Maintain the key store for the client certificate.

If the OAuth client shall authenticate via client ID and client certificate, then either upload a key store including the client certificate, the private key, and the certificate chain to the SAP Destination service, or generate one, for instance **my-keystore.p12**. For more information, see [Use Destination Certificates](#).

4. Create a communication user.

Create a communication user using the Maintain Communication Users app in your ABAP environment system. In this context the communication user represents the OAuth client. Enter the following data:

Field	Input
User Name	Provide a user name, for example, <b>MY_OAUTH_CLIENT</b> .
Description	Provide a description.

Field	Input
Password	If the OAuth client shall authenticate via client secret, provide a password.
Certificate	If the OAuth client shall authenticate via client certificate, then upload the client certificate to the communication user. During runtime, the service is processed by the communication user that is assigned to the certificate.

5. Create a communication system.

Create a communication system using the Communication Systems app in your ABAP environment system. Enter the following data:

General Data

Field	Input
System ID	Provide a system ID, for example, <b>MY_COMMUNICATION_PARTNER</b> .
System Name	Provide a system name.

Technical Data

Field	Input
General – Inbound Only	Yes

Identity Provider

Field	Input
OAuth 2.0 Identity Provider	On
User ID Mapping Mode	Select <i>User Name</i> . Thereby, if the NameldFormat is unspecified, then the value of the NameID element in the SAML assertions' subject will be mapped to the user name.
SAML Bearer Issuer	Enter the SAML entity ID of the SAML bearer issuer, which corresponds to the common name (the string after CN=) of the signing certificate subject.
Signing Certificate	Upload the signing certificate.

Users for Inbound Communication

Authentication Method	User Name/Client ID
OAuth 2.0	Add a user. User name of the communication user (= OAuth client ID), for example, <b>MY_OAUTH_CLIENT</b>

6. Create a communication arrangement.

Create a communication arrangement using the Communication Arrangements app in your ABAP environment system. Enter the following data:

Field	Input
Scenario ID	Select the scenario, for example, <b>Z_MY_SCENARIO</b> .
Arrangement Name	Provide a name, for example, <b>MY_ARRANGEMENT</b> .
Communication System	Select the communication system, for example, <b>MY_COMMUNICATION_PARTNER</b> .
Inbound Communication – User Name	Select the communication user, for example, <b>MY_OAUTH_CLIENT</b> .
Inbound Communication – Authentication Method	<b>OAuth 2.0</b>

7. Get technical data of the own communication system.

Open the own communication system in the Communication Systems app in your ABAP environment system. To open your own communication system, choose [Own SAP Cloud System](#).

Denote the following values:

- Host name, for example, `1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com`
- OAuth 2.0 confidential client token service URL, for example, `https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com/sap/bc/sec/oauth2/token`
- SAML2 audience, for example, `https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap-web.eu10.hana.ondemand.com`

8. Create a Destination

Create a destination using the Destinations editor in the SAP BTP cockpit. For more information, see [Using the Destination Editor in the Cockpit](#). Provide the following data:

Field	Input
Name	Enter the name of the destination, for example <b>my-oauth-saml-bearer-assertion-destination</b> .
Type	HTTP
URL	Enter <code>https://&lt;hostname of the own communication system&gt;</code> , for example, <code>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com</code> .
Proxy Type	Internet
Authentication	<b>OAuth2SAMLBearerAssertion</b>
Key Store Location	If you don't use the standard signing certificate, then select the name of the corresponding key store, for example <b>my-signing-keystore.p12</b> .
Key Store Password	Enter the password for the keystore.

Field	Input
Audience	Enter the SAML2 Audience from the own communication system, for example, <b>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap-web.eu10.hana.ondemand.com</b> .
AuthnContextClassRef	<b>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</b>
Use mTLS for token retrieval	Activate if the OAuth client shall be authenticated at the token endpoint using client certificate.
Client Key	Enter the user name of the communication user (= OAuth Client ID), for example, <b>MY_OAUTH_CLIENT</b> .
Token Service Key Store Location	If the OAuth client shall be authenticated at the token endpoint using client certificate, then select the name of the corresponding key store, for example <b>my-keystore.p12</b> .
Token Service Key Store Password	If the OAuth client shall be authenticated at the token endpoint using client certificate, then enter the password for the key store.
Token Service URL Type	Choose <b>Dedicated</b> .
Token Service URL	Enter the OAuth 2.0 confidential client token service URL from the own communication system, for example, <b>https://1a354373-d200-46f6-9d5c-daab9a65d9b6.abap.eu10.hana.ondemand.com/sap/bc/sec/oauth2/token</b> .
Token Service User	If the OAuth client shall be authenticated at the token endpoint using client ID and secret, then enter the name of communication user, for example, <b>MY_OAUTH_CLIENT</b> .
Token Service Password	If the OAuth client shall be authenticated at the token endpoint using client ID and secret, then enter the password of the communication user.
nameIdFormat	Enter <b>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</b> if the e-mail is propagated or <b>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</b> if the user name is propagated. See <a href="#">User Propagation via SAML 2.0 Bearer Assertion Flow</a> for more information.

## Related Information

[Maintain Communication Users \[page 2593\]](#)

[Communication Systems \[page 2598\]](#)

[Communication Arrangements \[page 2595\]](#)

### 6.3.2.3 Overview of Communication Scenarios Managed by SAP

Find a quick overview of all the communication scenarios in the ABAP environment.

#### Communication Scenarios

Communication Scenario	Documentation Link
SAP Enable Now Manager Integration (SAP_COM_0011)	<a href="#">Integrating SAP Enable Now Manager [page 2904]</a>
SAP Analytics Cloud Integration (SAP_COM_0065)	<a href="#">Integrating SAP Analytics Cloud [page 2872]</a>
Identity Management Integration (SAP_COM_0093)	<a href="#">Inbound Service: Business User [page 1048]</a>
SAP Cloud Identity Services - Identity Provisioning (SAP_COM_0193)	<a href="#">SAP BTP ABAP Environment</a>
SAP BTP Cloud Foundry Destination Service Integration (SAP_COM_0276).	<a href="#">Create a Destination [page 891]</a>
External Scheduler - Application Job Administration Integration	<a href="#">Integrating External Scheduler</a>
Business User Change Document Integration (SAP_COM_0327)	<a href="#">Integrating Business User Change Documents [page 1090]</a>
Business Role Change Document Integration (SAP_COM_0366)	<a href="#">Integrating Business Role Change Documents [page 1087]</a>
Application Monitoring Integration (SAP_COM_0454)	<a href="#">Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM [page 2956]</a>
Custom Code Migration Integration (SAP_COM_0464).	<a href="#">Enable Usage of the Custom Code Migration App [page 2611]</a>
Printing - Pull Integration (SAP_COM_0466)	<a href="#">Configuring the SAP Cloud Print Manager for Pull Integration</a>
Printing - Notification Integration (SAP_COM_0467)	<a href="#">Configuring the SAP Cloud Print Manager for Pull Integration</a>
Adobe Document Services (SAP Forms service by Adobe) Integration (SAP_COM_0503)	<a href="#">Print Forms [page 1620]</a>
SAP BTP ABAP Environment - Software Component Test Integration (SAP_COM_0510)	<a href="#">(Deprecated) Test Integration (SAP_COM_0510) [page 785]</a>
SAP Cloud ALM for Operations - Application Monitoring Push Integration (SAP_COM_0527)	<a href="#">Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM [page 2956]</a>
SAP BTP Workflow Integration (SAP_COM_0542).	<a href="#">Integrating Workflow [page 2919]</a>
E-Mail Integration (SAP_COM_0548).	<a href="#">Sending Mails Using SMTP [page 1032]</a>
SAP BTP ABAP Environment - Software Assembly Integration (SAP_COM_0582)	<a href="#">Software Assembly Integration (SAP_COM_0582) [page 795]</a>
SAP BTP Transport Management - Outbound Integration (SAP_COM_0599)	<a href="#">Manage Software Components</a>
UI Theme Designer Integration (SAP_COM_0623)	<a href="#">Integrating UI Theme Designer [page 2904]</a>
Communication Management - RFC Metadata Integration (SAP_COM_0636)	<a href="#">Develop a Remote-Enabled Function Module (RFM) [page 1040]</a>
FLP Content Exposure Integration (SAP_COM_0647)	<a href="#">Expose Launchpad Content to Launchpads on SAP BTP</a>

Communication Scenario	Documentation Link
Enabling Notifications (SAP_COM_0683)	<a href="#">Enabling Notifications on SAP BTP, ABAP environment in Launchpads Running on SAP BTP [page 2871]</a>
ABAP Test Cockpit - Test Integration (SAP_COM_0901)	<a href="#">Running ABAP Test Cockpit (ATC) Check Runs [page 770]</a>
ABAP Unit Test Integration (SAP_COM_0735)	<a href="#">Automate the Software Lifecycle Management Process</a>
Security Information and Event Management Integration (SIEM) (SAP_COM_0750)	<a href="#">Security Audit Log Integration</a>
ABAP Test Cockpit Configuration Integration (SAP_COM_0763)	<a href="#">Reading and Deploying ATC Configurations</a>
Factory Calendar - Pull Integration (SAP_COM_0834)	<a href="#">How to export using SAP Cloud Transport Management</a>

### 6.3.2.4 Integration Scenarios

You can integrate the ABAP environment with the following services.

[Integrating Security Audit Logs \[page 2864\]](#)

[Integrating Business Role Change Documents \[page 2868\]](#)

You can use this scenario to read the change documents of business roles.

[Integrating Business User Change Documents \[page 2869\]](#)

You can use this scenario to read the change documents of business users.

[Integrating Content with SAP Build Work Zone, standard edition \[page 2870\]](#)

Get an overview of the process for integrating content to the SAP Build Work Zone, standard edition.

This process is composed of a content exposure flow and a content consumption flow.

[Integrating SAP Analytics Cloud \[page 2872\]](#)

SAP Analytics Cloud is used as analytics client consuming ABAP environment data exposed via ABAP environment analytical queries.

[Integrating SQL Services Using SAP Datasphere \[page 2875\]](#)

You need the integration scenario SAP\_COM\_0532 if you want to create a replication flow in SAP Datasphere to subscribe to CDS view entities from an ABAP system and consume data from these views. With a communication arrangement based on SAP\_COM\_0532 combined with a communication arrangement for exposing the SQL service, you can consume data from custom CDS view entities that are exposed using the SQL service.

[Integrating SAP Document Management Service \[page 2877\]](#)

You can integrate the ABAP environment of SAP Business Technology Platform with SAP Document Management Service to establish a communication scenario.

[Integrating SAP BTP, ABAP environment and Local Printers \[page 2878\]](#)

To set up the integration between SAP BTP, ABAP environment and local printers, you can use two communication scenarios, SAP\_COM\_0466 and SAP\_COM\_0467.

[Integrating External Scheduler \[page 2883\]](#)

This inbound OData service enables external systems to schedule an application job based on an application job template.

[Integrating Factory Calendar \[page 2902\]](#)

Learn how to integrate an ABAP Platform system with the factory calendar.

#### [Integrating Outbound Emails Using SMTP \[page 2903\]](#)

Learn how to integrate outbound emails using the Simple Message Transfer Protocol (SMTP).

#### [Integrating SAP Enable Now Manager \[page 2904\]](#)

The ABAP environment supports the extended content scenario of SAP Enable Now that allows you to add an additional source for help content to the SAP Companion plugin. You can edit the context-sensitive in-app help delivered by SAP and adapt it to your needs or include your own content.

#### [Integrating UI Theme Designer \[page 2904\]](#)

To set up the integration between SAP BTP ABAP Environment and UI theme designer, you can use the communication scenario SAP\_COM\_0623.

#### [Integrating SAP Build Process Automation \[page 2909\]](#)

You can integrate your ABAP environment with SAP Build Process Automation.

#### [Integrating Workflow \[page 2919\]](#)

You can enable the communication between the ABAP environment and the workflow capability within the SAP Workflow Management service.

#### [Integrating Business Event Logging \[page 2925\]](#)

### 6.3.2.4.1 Integrating Security Audit Logs

Service name: RSAU\_LOG\_API

This service enables you to retrieve the security audit log data. You can use the audit log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

This is an OData version 4 service. This version aims to improve processing time and resource consumption of clients and servers. This includes a lightweight JSON format that reduces the size of every response.

#### Technical Details

A service group contains services belonging to the same business object model and so it shares similar environment conditions. This means the configuration and administration of a service group apply to all services in a service group, that is, routing, authorization, and so on, so you only have to do them once.

In the OData version 4 (V4) runtime implementation of the SAP Gateway Foundation, framework services originate from repositories.

Service Group (incl. Name-space if Existent)	Repository ID	Service Name (incl. Name-space if Existent)	Version
RSAU_LOG_API	SRVD_A2X	RSAU_LOG_API_SERVICE	0001

## Service Structure

### Service Header (optional)

The service header contains information about the service.

### Entities

The entities contain the business data of the service.

Entity	Description	Necessity	Link to Details
Security Audit Log	Security Audit Log		<a href="#">Retrieving Security Audit Log [page 2777]</a>

## Service Response

The details included in the response vary according to the type of operation. For more information, see Operation for Security Audit Log Integration.

## Integration with SIEM Solution

Service name: RSAU\_LOG\_API

Communication scenario: SAP\_COM\_0750

This service enables you to retrieve the security audit log data. You can use the audit log data to integrate them into your Security and Event Management solution (SIEM) to detect security relevant event situations.

## Additional Information

### ⓘ Note

For more details about Communication Management, see [Communication Management \[page 2591\]](#).

## 6.3.2.4.1.1 Operations for Security Audit Log Integration - Read

The Security Audit Log Integration - Read offers the following operations:

Operation	HTTP Method	Sample URL
<a href="#">Retrieving Security Audit Log [page 2777]</a>	GET	Example getting the first 1000 audit log entries:  GET <host>/sap/opu/odata4/sap/rsau_log_api/srvd_a2x/sap/rsau_log_api/0001/SecurityAuditLog?\$top=1000

### 6.3.2.4.1.1.1 Retrieving Security Audit Log

With `SecurityAuditLog` you can retrieve the Security Audit Log by using the HTTP method `GET`.

#### Request

You have to include the following properties in the URL of the request:

Property	Necessity	Comment
<code>eventID</code>	Optional	Security Audit Log Event <ul style="list-style-type: none"><li>• String &lt;= 3 characters Example: AU1, AU2</li></ul>
<code>log_tstmp</code>	Optional	Timestamp of the security audit log event <ul style="list-style-type: none"><li>• String(\$date-time) Example: 2017-04-13T15:51:04Z</li></ul>
<code>slgmand</code>	Optional	Client within the SAP System <ul style="list-style-type: none"><li>• String &lt;=3 characters Example: 100</li></ul>
<code>sid</code>	Optional	System ID <ul style="list-style-type: none"><li>• String &lt;=3 characters</li><li>• Example: YI3</li></ul>

Property	Necessity	Comment
counter	Optional	Message counter <ul style="list-style-type: none"> <li>• integer(\$int16)</li> </ul>
slgtc	Optional	Transaction Code <ul style="list-style-type: none"> <li>• String &lt;= 20 characters</li> </ul> Example: VA01
slgrepna	Optional	Program Name <ul style="list-style-type: none"> <li>• String &lt;= 40 characters</li> </ul> Example: SAPMSSY8

## Response

The operation returns the Security Audit Log.

## Examples

### Get events within a specific timeframe:

```
GET <host>/sap/opu/odata4/sap/rsau_log_api/srvd_a2x/sap/rsau_log_api/0001/
SecurityAuditLog?
$filter=(log_tstmp%20ge%202021-05-08T10%3A25%3A09Z%20and%20log_tstmp%20le%202021-
05-09T10%3A25%3A09Z)
```

## Response

```
{
  "eventID": "AU1",
  "log_tstmp": "2021-05-08T10:26:22.740611Z",
  "slgmand": "100",
  "sid": "ABC",
  "counter": 0,
  "terminal_name": "",
  "user_fullname": "Example Administrator",
  "slgtc": "S000",
  "slgrepna": "RSBTERTE",
  "rsau_text": "Logon successful (type=B, method=A)",
  "UserDescription": "Example Administrator"
},
{
  "eventID": "AU1",
  "log_tstmp": "2021-05-08T10:28:22.924215Z",
  "slgmand": "100",
  "sid": "ABC",
  "counter": 0,
  "terminal_name": "",
  "user_fullname": "Example Administrator",
  "slgtc": "S000",
```

```
"slgrepna": "RSBTERTE",
"rsau_text": "Logon successful (type=B, method=A)",
"UserDescription": "Example Administrator"
},
```

### 6.3.2.4.2 Integrating Business Role Change Documents

You can use this scenario to read the change documents of business roles.

#### Context

This scenario allows you to read the change documents of business roles. We recommend that you limit the number of change documents according to the *Changed On* and the *Business Role ID* fields to avoid a high volume of data.

#### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the *Maintain Communication Users* app.  
For more information, see the *Related information* section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the *Communication Systems* app.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business Role Change Document Integration* scenario (SAP\_COM\_0366) in the *Communication Arrangements* app.

For more information, see the *Related information* section.

#### Related Information

[How to Create a Communication Arrangement \[page 2596\]](#)

[How to Create Communication Users \[page 2594\]](#)

## 6.3.2.4.3 Integrating Business User Change Documents

You can use this scenario to read the change documents of business users.

### Context

This scenario allows you to read the change documents of business users. We recommend that you limit the number of change documents according to the *Changed On* and the *Business User ID* fields to avoid a high volume of data.

### Procedure

1. Check if a communication user already exists for this scenario. If not, create a communication user in the *Maintain Communication Users* app.  
For more information, see the *Related information* section.
2. Check if a communication system already exists for this scenario. If not, create a communication system in the *Communication Systems* app.  
For more information, see the *Related information* section.
3. Check if a communication arrangement already exists for this scenario. If not, create a communication arrangement for the *Business User Change Document Integration* scenario (SAP\_COM\_0327) in the *Communication Arrangements* app.  
For more information, see the *Related information* section.

### Related Information

[How to Create a Communication Arrangement \[page 2596\]](#)

[How to Create Communication Users \[page 2594\]](#)

## **6.3.2.4.4 Integrating Content with SAP Build Work Zone, standard edition**

Get an overview of the process for integrating content to the SAP Build Work Zone, standard edition. This process is composed of a content exposure flow and a content consumption flow.

### **Exposing Content to SAP Build Work Zone, standard edition**

Communication scenario SAP\_COM\_0647- SAP Fiori Launchpad Content Exposure Integration is needed to supply the SAP Build Work Zone, standard edition with content from the connected system.

For a description of the steps required to set up the communication scenario SAP\_COM\_0647, see [Expose Launchpad Content to Launchpads on SAP BTP](#).

### **Consuming Content in SAP Build Work Zone, standard edition**

#### **Prerequisite**

Launchpad content from the same system/client has been transferred to SAP Build Work Zone, standard edition using SAP\_COM\_0647 - SAP Fiori Launchpad Content Exposure Integration.

[Content Consumption - Cloud Solutions](#)

[Content Consumption - Portal Service](#)

[Content Consumption - SAP Work Zone](#)

#### **Related Information**

[Set SAP BTP, ABAP Environment as a Content Provider for SAP Build Work Zone, standard edition](#) 

## 6.3.2.4.4.1 Enabling Notifications on SAP BTP, ABAP environment in Launchpads Running on SAP BTP

Get an overview of the communication scenario SAP\_COM\_0683 - Enabling Notifications from SAP BTP, ABAP environment in launchpads running on SAP BTP. The launchpads running on SAP BTP include the SAP Build Work Zone, standard edition and SAP Cloud Portal service.

### Prerequisites

- To receive notifications, every user must have the same email configured in the SAP BTP system user management section and in the Identity Provider (IdP) used by the launchpad running on SAP BTP, Cloud Foundry environment.
- If you want to deliver push notifications to mobile devices, the target user is identified using the email ID they provided. Currently, it is not possible to use other types of identifiers.
- You've generated and noted the following parameters from the launchpad running on SAP BTP:
  - Host
  - OAuth 2.0 Client ID
  - Client Secret
  - Authorization Endpoint
  - Token Endpoint

### Procedure

1. Log on to the SAP Fiori launchpad in your SAP BTP, ABAP environment system.
2. Open the *Communication Arrangements* app.
3. Create a communication arrangement. Choose scenario SAP\_COM\_0683.
4. Enter a name for the communication arrangement.
5. Create a new communication system by choosing *New* next to the communication system.
6. Provide values for *System ID* and *System Name* and then choose *Create*.
7. In the *Technical Data* section, enter the value for *Host* obtained from the launchpad running on SAP BTP.
8. In the *OAuth 2.0 Settings* section, enter values for *Auth. Endpoint* and *Token Endpoint* obtained from the launchpad running on SAP BTP.
9. Create a new user for Outbound Communication by entering the following details:
  - Authentication Method: OAuth 2.0
  - Client Authentication: Basic
  - OAuth 2.0 Client ID: Client ID obtained from the launchpad running on SAP BTP
  - Client Secret: Client Secret obtained from the launchpad running on SAP BTP
10. Save the Communication System and Communication Arrangement.

## Related Information

[Working with Notifications](#)

### 6.3.2.4.5 Integrating SAP Analytics Cloud

SAP Analytics Cloud is used as analytics client consuming ABAP environment data exposed via ABAP environment analytical queries.

#### Prerequisites

- You have the *BI Admin* role in the SAP Analytics Cloud system. For more information, see [Requesting Roles](#).
- You have the *Administrator* role (role template ID `SAP_BR_ADMINISTRATOR`). For more information, see [Maintain Business Roles](#).
- Your end users' web browsers must be configured to:
  - Allow pop-up windows from the SAP Analytics Cloud domain: `[ * . ]sapbusinessobjects.cloud`, if you'll be choosing the single sign-on (SSO) authentication method in step 2.
  - Allow 3rd party cookies from the Tomcat server's domain. For example, in Internet Explorer 11, go to  *Internet Options*  *Security*  *Trusted Sites* , add your domain name, then select *Enable Protected Mode*.

#### Context

You can configure a live data connection in SAP Analytics Cloud and create an SAP Analytics Cloud model based on an ABAP environment analytical query to consume the result set in an SAP Analytics Cloud Story. The analytical query is delivered as CDS content and is exposed via the analytical engine and the InA protocol via the REST services under `/sap/bw/ina/...` to SAP Analytics Cloud.

SAP BTP destinations are created by the SAP Analytics Cloud app when you define an SAP Analytics Cloud connection of the type ABAP environment.

You set up the scenario using the following tasks:

1. [Configure Identity Authentication \[page 2873\]](#)
2. [Create a Communication System \[page 2873\]](#)
3. [Create a Communication Arrangement \[page 2874\]](#)
4. [Configure Live Data Connection \[page 2875\]](#)

## 6.3.2.4.5.1 Configure Identity Authentication

Configure your Identity Authentication tenant as the identity provider.

### Prerequisites

You have the *System Owner* role.

### Procedure

1. Log on to the SAP Analytics Cloud system via the default identity provider tenant.  
The identity provider is the Identity Authentication tenant that you get with the ABAP environment tenant.
2. Go to ► *System* ► *Administration* ► *Security* ▾ and change the *Authentication Method* from *SAP Cloud Identity (Default)* to *SAML Single Sign-On (SSO)*, then follow the steps for *SAML Single Sign-On (SSO Configuration)*.
3. Under *Step 2: Upload Identity Provider metadata*, deselect *Identity Provider will also be used...* if this checkbox is visible.
4. Under ► *Step 3* ▾, set the *SAML User Mapping* to *Custom SAML User Mapping*. Under ► *Security* ► *Users* ▾, the value in the *Custom SAML User Mapping* column must equal the ► *User Data* ► *User Name* ▾ of the corresponding business user in the ABAP environment.

## 6.3.2.4.5.2 Create a Communication System

You can create a communication system in the ABAP environment.

### Prerequisites

You have the *Administrator* role (role template ID `SAP_BR_ADMINISTRATOR`).

### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment.
2. Open the *Communication Systems* app from the SAP Fiori Launchpad.
3. Choose *New*.

4. Enter *System ID* and *System Name*. The system ID should be unique.
5. Turn off the *Destination Service* to view the *Host Name*.
6. Under [ ], enter the *Host Name* of the SAP Analytics Cloud tenant and *Port* as **443**. SAP Analytics Cloud follows the host name schema `xxx.sapanalytics.cloud`.
7. Choose *Save*.

### 6.3.2.4.5.3 Create a Communication Arrangement

You create a communication arrangement in the ABAP environment.

#### Prerequisites

You have the *Administrator* role (role template ID `SAP_BR_ADMINISTRATOR`).

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment.
2. Open the *Communication Arrangement* app.
3. Choose *New*.
4. Select the communication scenario `SAP_COM_0065`.
5. Enter a name for the arrangement.
6. Choose *Create*.
7. In the *Common Data* section, select the *Communication System* that you created in the previous section.
8. Under [ ] maintain the tenant ID that is visible in the URL of the SAP Analytics Cloud app.

When you call `https://xxx.sapanalytics.cloud`, you will be redirected to the full URL where you can find the tenant ID as shown in this example: `https://xxx.sapanalytics.cloud/sap/fpa/ui/tenants/<tenant_ID>/app.html`. You can also find the tenant ID under [ ] [ ] [ ].

9. Under [ ], check the *Service Status* as *Active*.
10. Under *Retrieve Stories*, uncheck the *Service Status*.

#### Note

Retrieving stories is not supported.

11. Choose *Save*.

#### Note

Do not create more than one communication arrangement with communication scenario SAP\_COM\_0065 in your ABAP Environment tenant.

### 6.3.2.4.5.4 Configure Live Data Connection

You configure a live data connection in SAP Analytics Cloud.

#### Prerequisites

You have the *BI Admin* role in the SAP Analytics Cloud system.

#### Procedure

1. Login to the SAP Analytics Cloud tenant.
2. Choose   .
3. In the *Select a data source* dialog, expand *Connect to Live Data* and select *SAP S/4 HANA*.
4. In the *S/4 HANA Live Connection* dialog, enter the name and description.
5. Select the *Connection Type* as **Direct Connection**.
6. Enter the host name as `xxx.abap-web.stagingaws.hanavlab.ondemand.com`.
7. Enter *HTTPS Port* as **443** and *Client* as **100**.
8. Select the *Authentication Method* as **SAML Single sign-on (Standard Compliant)**.
9. Choose *OK*.

### 6.3.2.4.6 Integrating SQL Services Using SAP Datasphere

You need the integration scenario SAP\_COM\_0532 if you want to create a replication flow in SAP Datasphere to subscribe to CDS view entities from an ABAP system and consume data from these views. With a communication arrangement based on SAP\_COM\_0532 combined with a communication arrangement for exposing the SQL service, you can consume data from custom CDS view entities that are exposed using the SQL service.

The connection that you establish for this scenario is shared by all SAP Datasphere users, and SAP Datasphere controls the usage of the connection and accessed data assets through its integrated policy management agent.

SAP Datasphere connects to the CDC (Change Data Capture) functionality for CDS view entities that records changes of exposed CDS view entities. This component allows multiple consumers to subscribe to a CDS view entity to get the CDS data including the option for delta recording and delta consumption.

SAP Datasphere either adds another subscription to an existing CDS extraction process (if the CDS view entity is already processed by another subscriber) or creates a new extraction process and adds a subscription (if the CDS view entity is configured for extraction the first time), followed by the consumption of the data.

## Prerequisites

- You have entered the user and password for your user (typically a technical user) and have selected [SAP ABAP](#) as the connection type for the connection to the SAP system in the [Connections](#) area.
- The user in the SAP system that you use to complete the configuration steps should have administrator privileges or at least Business Catalog ID `SAP_CORE_BC_COM` for access to [Communication Management](#).
- You are familiar with steps required to expose CDS view entities. For more information, see [Data Consumption Using SAP Datasphere \[page 1203\]](#).

## Procedure

1. Create a communication user in the [Maintain Communication Users](#) app. You will later need this user name and password to establish the connection in SAP Datasphere.
2. Create a communication system in the [Communication Systems](#) app and select the checkbox [\*Inbound only\*](#).  
**Background information:** A communication system is important for outbound services, as it is used to register a destination for the communication system in the ABAP system. In the given scenario, requests always originate from the external system, so selecting the checkbox [\*Inbound only\*](#) is sufficient.
3. For a production scenario, switch on [OAuth 2.0 Identity Provider](#), enter a provider name and upload a signing certificate. Note that this step is not relevant if you are creating a test scenario.
4. Add the communication user that you created in step 1 with authentication method [\*User Name and Password\*](#) to [\*Users for Inbound Communication\*](#).
5. Create a communication arrangement for communication scenario `SAP_COM_0532` in the [Communication Arrangement](#) app and enter the name of the communication system that you created in step 2. Note that you do not need to specify an authorization group ID.
6. In SAP Datasphere, choose [Connections](#). On the [Local Connections](#) tab, choose the [Create](#) button.
7. Enter the necessary information. Choose the connection type [SAP ABAP](#). In the [Protocol](#) field, select the value [Web Socket RFC](#). In the [SAP Logon Connection Type](#) field, select the value [Application Server](#). In the [Use Cloud Connector](#) field, ensure the value [False](#) is selected. Use the inbound user data from step 4.
8. Test the connection by choosing the [Validate](#) button. Ensure that the status is [OK](#) before proceeding.
9. Create a communication arrangement based on a communication scenario with object privileges so that the communication user created in step 1 can access the exposed CDS view entities. For more information, see [Creating a Communication Arrangement for Exposing the SQL Service \[page 1216\]](#).

You can now create a replication flow in SAP Datasphere to consume data from CDS view entities.

## 6.3.2.4.7 Integrating SAP Document Management Service

You can integrate the ABAP environment of SAP Business Technology Platform with SAP Document Management Service to establish a communication scenario.

### Prerequisites

- You've subscribed to the service Document Management Service, Integration Option and created an instance in Cloud Foundry environment. See [Initial Setup for Document Management Service, Integration Option](#).
- The global administrator of your SAP BTP account has added the service plan for Document Management Service, Repository Option to your subaccount via entitlements. See [Configure Entitlements for Subaccounts](#).
- You've onboarded Document Management provided storage capability. See [Connect to Document Management Service, Repository Option Using API](#).
- You've created and copied the service key in the same subaccount of Cloud Foundry environment where Document Management, integration option instance is created. For more information about creating service key, see [Create Service Keys Using the Cockpit](#).
- You've the *Administrator* role (role template ID `SAP_BR_ADMINISTRATOR`). See [Maintain Business Roles](#).

### Context

SAP Document Management Service helps companies with managing records or attachments that are linked to business objects or a specific business context.

For each tenant of SAP BTP ABAP environment, a service instance of SAP Document Management Service in Cloud Foundry environment has to be created.

You can build and edit communication arrangements that your organization has set up with a communication partner using the *Communication Arrangement* application. The system provides communication scenarios for inbound and outbound communication that you can use to create communication arrangements. Inbound communication describes how a communication partner receives business documents, while outbound communication describes how a communication partner sends business documents to a communication partner. The communication scenario specifies the authorizations, inbound and outbound services, and supported authentication methods that are necessary for the communication.

#### Note

For the collection of sample classes available in the following GIT Hub location <https://github.com/SAP-samples/abap-cmis-client-sample> that helps you to use SAP Document Management Service, Integration option, in the SAP BTP, ABAP environment.

## Related Information

[Create a Communication Arrangement \[page 2878\]](#)

### 6.3.2.4.7.1 Create a Communication Arrangement

Create a communication arrangement in the ABAP environment.

#### Prerequisites

You've the *Administrator* role (role template ID `SAP_BR_ADMINISTRATOR`).

#### Procedure

1. Log on to the SAP Fiori launchpad in the ABAP environment.
2. Open the *Communication Arrangement* app from the SAP Fiori Launchpad.
3. Choose *New*.
4. Select the communication scenario `SAP_COM_0668`.
5. Enter *Service Key* that you generated. For more information about creating service key, see [Create Service Keys Using the Cockpit](#).
6. Choose *Create*.

A communication system created with the name `ZSAP_COM_0668`.

#### ⓘ Note

Don't create more than one communication arrangement with communication scenario `SAP_COM_0668` in your tenant in the ABAP environment.

### 6.3.2.4.8 Integrating SAP BTP, ABAP environment and Local Printers

To set up the integration between SAP BTP, ABAP environment and local printers, you can use two communication scenarios, `SAP_COM_0466` and `SAP_COM_0467`.

Both communication scenarios are designed to connect third-party output management systems (OMS). OMS provide you with more refined printing functionalities. With `SAP_COM_0466`, the OMS use polling to check for the availability of new items to print. The communication scenario only has inbound communication. The [SAP Cloud Print Manager for Pull Integration](#) is supported by this communication scenario.

With SAP\_COM\_0467, the SAP BTP, ABAP environment system sends a notification to the OMS if new items are available to print. Therefore, the communication scenario has in- and outbound communication. This communication scenario doesn't support the use of the *SAP Cloud Print Manager for Pull Integration*.

Item retrieval and status update is the same for both methods. The only difference is the determination of new items to print.

Please check the documentation of your OMS first to see which method is supported before creating a communication arrangement of any type. An example of an implementation of these communication scenarios is Microsoft Universal Print. See [this blog](#) for more details.

## Related Information

[SAP\\_COM\\_0466 \[page 2879\]](#)

[SAP\\_COM\\_0467 \[page 2881\]](#)

### 6.3.2.4.8.1 SAP\_COM\_0466

This document describes the configuration steps that have to be carried out by customers to set up the integration between SAP BTP, ABAP environment and local printers using SAP\_COM\_0466.

## Purpose

The communication scenario SAP\_COM\_0466 enables you to print from an system on your locally installed printers. The communication scenario is designed to connect the *SAP Cloud Print Manager for Pull Integration* and third-party output management systems (OMS). OMS provide you with more refined printing functionalities.

### ⓘ Note

The term 'OMS' will stand for both an OMS as well as the *SAP Cloud Print Manager for Pull Integration* for the remainder of this document.

## Prerequisites

- You've installed an OMS.
- You have the following business role assigned to your user:
  - SAP\_BR\_ADMINISTRATOR
- This business role contains the following business catalogs:
  - Communication Management (SAP\_CORE\_BC\_COM)

- Output Management (SAP\_CORE\_BC\_OM\_PRT)

## Process Steps

### (SAP BTP, ABAP environment)

1. Log in to your system and navigate to the *Display Communication Scenarios* app.
2. Search for scenario ID SAP\_COM\_0466 and click on it.
3. Now click on *Create Communication Arrangement* in the top right corner.
4. Choose a name for your communication arrangement and click *Create*.
5. You will now need to create a communication system. You can do this by clicking *New* next to the field *Communication System*. For more information on general concepts of communication management, see [Communication Management \[page 2591\]](#).
6. Choose a new *System ID* and click *Create*.

#### ⓘ Note

The *System Name* is filled in automatically with the same name as the ID. We recommend leaving this as it is to prevent confusion.

7. In the *General* section, mark the *Inbound Only* checkbox.
8. Now scroll down to *Users for Inbound Communication* and click "+" to add a new communication user.
9. Click on *New User*. Enter a username and a description. You can then choose to either enter a new password or upload a client certificate by clicking on *Upload*. Click *Create* to save the user, then click *OK*.
10. Save your communication system by clicking *Save* on the bottom right, then save your communication arrangement the same way.
11. Now search for the *Maintain Print Queues* app, open it and select *New* to create a print queue of the type *Pull Integration Queue*.

#### ⓘ Note

The selection of the print queue type is only possible if more types are available. If this is not the case, *Pull Integration* is automatically selected. In case more options are available, it is important to select the *Pull Integration* queue type for the scenario. You won't see the print queue in your OMS system if you choose the wrong queue type.

12. Enter a name and description for the queue. Choose a format and make sure to select the name you chose for your communication arrangement in step 3. Leave the other settings as they are and save your test queue by clicking *Create*.
13. Scroll to the bottom of the print queue list window and select *System URL*. Copy the URL, as you'll need it later on.
14. Open the print queue and create a couple of test pages by clicking *Create Test Page*. They'll have the status *New*.
15. **(OMS):** Open your OMS and log in to the SAP BTP, ABAP environment system by using the system URL you copied in step 13 as well as the user and password created in step 8 and assign a physical printer to your print queue. Now you can print the test pages from your OMS. For details, please refer to the individual product documentation.
16. **(SAP Cloud Print Manager for Pull Integration):** For more information on how to configure your SAP Cloud Print Manager for Pull Integration, see [Communication Management \[page 883\]](#).

17. (**SAP BTP, ABAP environment**) : In the *Maintain Print Queues* app, your documents will now be labeled *Successful*.

## 6.3.2.4.8.2 SAP\_COM\_0467

This document describes the configuration steps that have to be carried out by customers to set up the integration between SAP BTP, ABAP environment and local printers using SAP\_COM\_0467.

### Purpose

The communication scenario SAP\_COM\_0467 enables you to print from an SAP BTP, ABAP environment system on your locally installed printers. The scenario consists of an inbound OData API which is designed for third-party output management systems (OMS) to connect, as well as a REST outbound API for the notification of new documents to be printed. OMS provide you with more refined printing functionalities.

### Prerequisites

- You've installed an OMS.
- You've configured a user in your OMS.
- You have the following business role assigned to your user:
  - SAP\_BR\_ADMINISTRATOR
- This business role contains the following business catalogs:
  - Communication Management (SAP\_CORE\_BC\_COM)
  - Output Management (SAP\_CORE\_BC\_OM\_PRT)

### Process Steps

1. Log in to your SAP S/4HANA system. Go to the *Communication Management* section and start the app **Display Communication Scenarios**.
2. Search for scenario ID SAP\_COM\_0467 and click on it.
3. Now click on *Create Communication Arrangement* in the top right corner.
4. Choose a name for your communication arrangement and click *Create*.
5. You will now need to create a communication system. You can do this by clicking *New* next to the field *Communication System*.
6. Choose a new *System ID* and click *Create*.

#### ⓘ Note

The *System Name* is filled in automatically with the same name as the ID. We recommend leaving this as it is to prevent confusion.

7. In the *General* section enter the host name, the IP address, or the URL of your OMS. Leave the HTTPS port as it is.
8. Now scroll down to *Users for Inbound Communication* and click "+" to add a new communication user.
9. Click on *New User*. Enter a username and a description. You can then choose to either enter a new password or upload a client certificate by clicking on *Upload*. Click *Create* to save the user, then click *OK*.
10. Now scroll down to *Users for Outbound Communication* and click "+" to add a new communication user.
11. Enter the username and password you created in your OMS. Click *Create* to save the user, then click *OK*.
12. Save your communication system by clicking *Save* on the bottom right.
13. Scroll down to the *Outbound Services* section and enter the service path you got from your OMS provider in the field *Path*. The service url will be configured automatically.
14. Save your communication arrangement by clicking *Save* on the bottom right.
15. Now search for the *Maintain Print Queues* app, open it and select *New* to create a print queue of the type **OMS for Notification Integration Queue**.
16. Enter a name and a description for the queue. Choose a format and make sure to select the name you chose for your communication arrangement in step 4. Set the notification type: When print queue items are added to the queue, a notification call is sent out. Here you can decide whether you want notifications for a certain number of items or for every single new item. Leave the other settings as they are and save your test queue by clicking *Create*.
17. Scroll to the bottom of the print queue list window and copy the system URL, as you'll need it later on.
18. Open the test queue and create a couple of test pages by clicking *Create Test Page*. They'll have the status *New*.
19. **(OMS):** Open your OMS and log in to the S/4 HANA system by using the system URL you copied in step 17 as well as the user and password created in step 9 and assign a physical printer to your print queue. Now you can print the test pages from your OMS. For details, please refer to the individual product documentation.
20. **(SAP BTP, ABAP environment):** In the *Maintain Print Queues* app, your documents will now be labeled *Successful*.

### 6.3.2.4.8.3 Managing Print Profiles

You can manage your print profiles for SAP\_COM\_0466 and SAP\_COM\_0467 in the *Maintain Print Queues* app.

#### Procedure

1. Assign a printer to a 0466 or 0467 print queue in your OMS or in the *SAP Cloud Print Manager for Pull Integration*. Mind that print profiles in the print manager are only supported when in PDF format. Make sure to ignore the print profile in the *PDF Settings* of your print manager in order to secure a correct runtime of the tool.

#### Note

If you've used SAP\_COM\_0467 to integrate your local printers to the SAP BTP, ABAP environment system, you can only choose a local printer from your external OMS instead.

- Now, open the *Maintain Print Queues* app and select the print queue for which you want to define your print profiles.
- Select *Manage Print Profiles* in the taskbar to set your print profile.
- In the *Print Options* dialog, you can manage your print settings. After you've selected your individual print settings from the dropdown menu, select *Save As*.
- Choose a profile name. Select *Use as favorite* to save the settings of your print profile. If no favorite has been selected, a default will be used. Each print job going into the same print queue will use these same settings. Click *OK*. Your new print profile has now been created. To delete a print profile, select *Delete* from the *Print Options* dialog.

 **Note**

You can only define print profiles if you have administrator rights.

## Related Information

[SAP\\_COM\\_0466 \[page 2879\]](#)

[SAP\\_COM\\_0467 \[page 2881\]](#)

### 6.3.2.4.9 Integrating External Scheduler

This inbound OData service enables external systems to schedule an application job based on an application job template.

The technical name of this inbound OData service is BC\_EXT\_APPJOB\_MANAGEMENT. Please see the description of the operation *JobSchedule* for restrictions. Additionally, the service offers other elementary operations on application jobs.

## Service Structure

### Entities

The entities contain the business data of the service.

Entity	Description
Jobinfo	Detailed information on a single application job.
JobTemplate	Information on an application job template.
JobScheduleStatus	Information on a scheduled job.
FuncImpReturn	Return code.

## Service Response

The details included in the response vary according to the type of operation. For more information, see [Operations for External Job Scheduling Service \[page 2884\]](#).

### 6.3.2.4.9.1 Operations for External Job Scheduling Service

The external job scheduling service API offers the following operations:

Operation	HTTP Method	Sample URL
<a href="#">Read an Application Job Without Parameters [page 2889]</a>	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobHeaderSet(JobName='JobName',JobRunCount='Job-Count')?\$expand=JobStepSet/JobStepLogInfoSet/JobLogMessageSet
<a href="#">Read Application Job Templates [page 2890]</a>	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobTemplateSet
<a href="#">Schedule an Application Job [page 2891]</a>	POST	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobSchedule?JobTemplateName='TemplateName'&Job-Text='Test'&JobUser='BusinessUser'&TestModelInd=false&JobParameterValues=""
<a href="#">Read Application Job Status [page 2892]</a>	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobStatusGet?JobName='JobName'&JobRunCount='Job-Count'
<a href="#">Abort a Running Application Job [page 2893]</a>	POST	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobAbort?JobName='JobName'&JobRunCount='Job-Count'
<a href="#">Cancel an Application Job [page 2894]</a>	POST	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobCancel?JobName='JobName'&JobRunCount='Job-Count'
<a href="#">Read Detailed Job and Step Information [page 2895]</a>	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobInfoGet?JobName='JobName'&JobRunCount='Job-Count'

Operation	HTTP Method	Sample URL
Read Application Job Status Texts [page 2896]	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobStatusInfoSet
Restart a Failed Application Job [page 2897]	POST	<host>/sap/opu/odata/sap/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobRestart?JobName='JobName'&JobRunCount='JobCount'&JobRestartMode='E'
Check Parameters for Scheduling a Job [page 2898]	POST	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobScheduleCheck?JobTemplateName='TemplateName'&JobParameterValues='ParameterValueString'
Retrieve Application Job Parameters [page 2899]	GET	<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobParamValues-Get?JobName='JobName'&JobCount='JobCount' and <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGE-MENT;v=0002/JobParamValues-StructGet?JobName='JobName'&JobCount='JobCount'

For more information on these operations, see [3383044](#).

### Working with \$Batch

The OData implementation can optionally support running multiple operations sent in a single HTTP request through the use of batching. This is especially useful when scheduling application jobs with a large amount of application job parameters, as these parameters are added to the request URL and may exceed the maximum number of allowed characters. For details, see the chapter **Batch** on page 17 of [3383044](#).

### Error Handling

In case of an error, the external job scheduling service API sets the http status code to 400 and returns a message of an SAP message class. In the http response, you'll see the message class, the message number, and the message text. One example:

#### Sample Code

```
<code>APJ_RT/003</code><message xml:lang="en">Job doesn't exist.</message>
```

In this example, the message with number 003 of message class APJ\_RT and the text "Job doesn't exist" was returned.

In most cases, a message of the message class APJ\_RT is used. The following table lists the message numbers and the corresponding texts:

Message code	Message constant	Text
001	TECHNICAL_ERROR	Job was aborted due to a technical error.
002	no_appl_log_found	No application log found
003	no_job_exists	Job doesn't exist.
004	cancel_job_failed	Job &1 couldn't be canceled.
005	no_cancel_privilege	User &1 has no authorization to cancel job &2.
006	delete_job_failed	Job &1 couldn't be deleted.
007	no_job_delete_privilege	User &1 has no authorization to delete job &2.
008	job_cancel_internal_error	Job &1 couldn't be canceled due to an internal error.
009	job_cancellation_succesful	Job &1 has been canceled.
010	job_deletion_succesful	Job &1 has been deleted.
011	job_wrong_state_for_deletion	You can't delete job &1 due to its status.
012	no_runs_for_job_found	Job &1 has no runs.
013	job_schedule_successfully	Job &1 has been scheduled.
014	job_catalog_entry_not_found	Job catalog entry &1 doesn't exist.
015	job_template_entry_not_found	Job template &1 doesn't exist.
016	report_for_job_not_found	Job &1 couldn't be created.
017	job_creation_failed	Report for job &1 couldn't be found.
018	job_metadata_inconsistent	Job &1 metadata are inconsistent.
019	read_job_log_failed	Log for job &1 couldn't be read.
020	job_start_immediate_failed	Job &1 couldn't start immediately
021	job_invalid_startdate	Job &1 has an invalid start date.
022	job_close_failed	Job &1 couldn't be scheduled.
023	job_list_internal_error	Internal error occurred during job list processing.
024	user_no_auth_report	User &1 not authorized to schedule report &2. Report on BC-SRV-APS-APJ.
025	future_schedules_restricted	Job &1 was restricted to 1000 scheduled runs.
026	job_completed_wo_info	Job completed without errors
027	no_auth_appl_log	User &1 is not authorized to read the log.

Message code	Message constant	Text
028	no_auth_job_details	User &1 is not authorized to read the job details.
029	no_auth_job_log	User &1 is not authorized to read the job log.
030	no_auth_job_delete	User &1 is not authorized to delete the job.
031	no_auth_job_cancel	User &1 is not authorized to cancel the job.
032	job_scheduling_failed	Job couldn't be scheduled.
033	job_no_end_date	No end date supplied for job &1.
034	job_start_before_end_date	Job &1 must start before the end date.
035	job_no_max_iterations	Enter a number of iterations for the job &1.
036	job_end_cond_ignored	End date of job &1 will be ignored due to missing recurrence pattern.
037	job_aborted	An error occurred. Please create an incident on component &1.
038	job_cancel_checking_failed	Job &1 couldn't be canceled: Check of job failed
039	job_cancel_not_exist	Job couldn't be canceled: Job does not exist.
040	JOB_CANCEL_NOT_RUNNING	Job couldn't be canceled: Job is not running.
041	job_delete_event_failed	Job &1 couldn't be deleted: Event scheduling can't be deleted
042	job_delete_steps_failed	Job &1 couldn't be deleted: Job steps couldn't be deleted
043	job_delete_time_failed	Job &1 couldn't be deleted: Deadline scheduling can't be deleted
044	job_delete_derelease	Job &1 couldn't be deleted: Previous job can't be modified
045	job_delete_enq_pre	Job &1 couldn't be deleted: Succeeding job can't be locked
046	job_delete_enq_suc	Job &1 couldn't be deleted: Previous job can't be locked
047	job_delete_enq_tbtco	Job &1 couldn't be deleted: Job can't be locked
048	job_delete_update_pre	Job &1 couldn't be deleted: Previous job can't be modified
049	job_delete_update_suc	Job &1 couldn't be deleted: Succeeding job can't be modified

Message code	Message constant	Text
050	job_delete_commit_failed	Job &1 couldn't be deleted: A technical error occurred.
051	job_delete_already_run	Job &1 couldn't be deleted: Job has already been published.
052	illegal_report_or_variant	Report name or variant name contains invalid characters.
053	illegal_variantname	Variant couldn't be created: Its name is empty, or contains '&' or '%'.
054	no_auth_variant_create	User &1 is not authorized to create a variant.
055	variant_not_executed	Variant couldn't be created: Variant can't be executed.
056	variant_no_report_exists	Variant couldn't be created: Report doesn't exist.
057	variant_no_report_supplied	Variant couldn't be created: Report is not of type 1.
058	variant_locked	Variant couldn't be created: Variant is locked.
059	variant_lock_enqueue_error	Variant couldn't be created: Internal error from enqueue server
060	job_must_not_periodic	Job &1 must not be periodic. Report incident on component BC-SRV-APS-APJ.
061	job_wrong_periodic_value	Value &1 for recurrence pattern is invalid for job &2.
062	invalid_factory_calendar	Job &1 is not scheduled due to invalid calendar.
063	invalid_restriction_code	Job &1 is not scheduled due to invalid restriction code
064	job_list_filter_failed	Filter for job list couldn't be applied.
065	job_delete_internal_error	Job couldn't be deleted due to an internal error.
066	INVALID_MONTH_SHIFT_DIR	Invalid shift direction for days of month
067	WEEKDAY_WRONG_STARTDATE	Start date and day of recurrence must be the same weekday.
068	INVALID_TIMEZONE	Time zone &1 is invalid.
069	INVALID_FC_ID	Calendar &1 is invalid.
070	SCHEDULING_DATE_ADJUSTED	Start date was adjusted to fit the recurrence pattern.
071	INVALID_SCHEDULING_START_TS	Job must start before the end date.
072	SELECT_AT_LEAST_ONE_WEEKDAY	For weekly recurrence, select at least one weekday.

Message code	Message constant	Text
073	CREATE_JOB_VARIANT_FAILED	Job variant couldn't be created.
074	VAL-UES_OF_READ_ONLY_PARAM_REM	Parameter &1 has been removed because it is read-only
075	USER_NO_AUTH_TO_SCHEDULE_JOB	Parameter &1 has been removed because it's read-only
076	JOB_CANCELED_BY_USER	Job was canceled by user &1.
077	NO_AUTH_SCHEDULE_FOR_USER	User &1 has no authorization to schedule the job.
078	PARAM_CHANGE_BEFORE_SCHEDULE	Invalid change of parameter when scheduling. Please open a ticket on &1
079	INVALID_JOB_RUN_STATUS	Invalid job run status.
080	JOBCATALOG_IS_NOT_BASIC	Job catalog is not basic.
081	JOB_RESTART_INCORRECT_STATUS	Can't restart the job due to the status
082	JOB_RESTART_INCORRECT_MODE	Can't restart the job after the last step
083	INVALID_USER	User &1 doesn't exist or is invalid.

### 6.3.2.4.9.1.1 Read an Application Job Without Parameters

Read an application job including step and log data, without parameters, using the HTTP method `GET`. An application job consists of a job header, a job step, a log header, and a log message. Via `$expand` all these job data can be read with one request.

#### Request

You can retrieve all entities of an application job

```
with one call: GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/
JobHeaderSet(JobName='JobName',JobRunCount='JobCount')?$expand=JobStepSet/
JobStepLogInfoSet/JobLogMessageSet
```

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobName	Mandatory	This is the name of the application job.
JobRunCount	Mandatory	This is the ID of the application job.

## Example

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/
JobHeaderSet(JobName='JobName',JobRunCount='JobCount')?$expand=JobStepSet/
JobStepLogInfoSet/JobLogMessageSet
```

For more information, see [3383044](#).

### 6.3.2.4.9.1.2 Read Application Job Templates

Get a list of all application job templates using the HTTP methode GET.

#### Request

Use the URL <host>/sap/opu/odata/SAP/BC\_EXT\_APPJOB\_MANAGEMENT;v=0002/JobTemplateSet without parameters.

#### Response

The operation returns a list of all application job templates.

## Examples

#### Request

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobTemplateSet
```

#### Response

```
{
  "d": {
    "results": [
      {
        "JobTemplateName": "string",
        "JobTemplateVersion": "string",
        "JobTemplateStepCount": "Unknown Type: integer,null",
        "JobPeriodicGranularity": "Unknown Type: string,null",
        "JobReportName": "Unknown Type: string,null",
        "JobUserName": "Unknown Type: string,null",
        "JobPeriodicValue": "Unknown Type: string,null",
        "JobTemplateText": "Unknown Type: string,null",
        "CreationDateTime": "/Date(1492098664000)/",
        "CreationUserName": "Unknown Type: string,null",
        "LastChangeDateTime": "/Date(1492098664000)/",
        "JobTemplateStepCount": 0
      }
    ]
  }
}
```

```

        "LastChangeUserName": "Unknown Type: string,null",
        "SupportsTestModeInd": "Unknown Type: boolean,null"
    }
}
}
}

```

### 6.3.2.4.9.1.3 Schedule an Application Job

With `JobSchedule`, you can schedule an application job based on a job template using the HTTP method `POST`. No start condition can be specified, it's always an immediate start.

#### Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
TestModelInd	Optional	Reserved should be set to false.
JobParameterValues	Optional	These are the parameter values of the application job.
JobParameterValuesSimple	Optional	This is the parameter string containing the step number. You can schedule application jobs with a simplified format, containing the step number, the application job name, and the <code>T_VALUE</code> .
JobTemplateName	Mandatory	This is the name of the application job template.
JobText	Mandatory	This is the name of the application.
JobUser	Mandatory	This is the user ID of the user performing the job.
JobUserName	Obsolete	This is the user name of the user performing the job.
JobUserID	Obsolete	This is the ID of the application job user.

#### Response

The operation schedules an application job.

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobSchedule?  
JobTemplateName='TemplateName'&JobText='Test'&JobUser='BusinessUser'&TestModeInd=  
false&JobParameterValues=''
```

### Response

```
{  
  "d": {  
    "results": [  
      {  
        "JobName": "string"  
        "JobRunCount": "string"  
        "JobStatus": "char"  
        "ReturnCode": "integer" (0 = ok, <> 0 = error)  
      }  
    ]  
  }  
}
```

### Example of JobParameterValuesSimple

#### ↳ Sample Code

```
'{ \"VALUES\" :[ { \"STEP_NR\" :StepNum , \"NAME\" :\"ParameterName\" , \"T_VALUE\" :  
[ { \"SIGN\" :\"I\" , \"OPTION\" :\"EQ\" , \"LOW\" :\"Value or X depending  
on parameter Type\" , \"HIGH\" :\" \"} ] } ] }'
```

## 6.3.2.4.9.1.4 Read Application Job Status

Read the status of an application job using the HTTP method GET.

### Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobRunCount	Mandatory	This is the ID of an application job.
JobName	Mandatory	This is the name of an application job.

## Response

The operation returns the status of an application job.

## Additional Information

Please note that to obtain the status of jobs that weren't scheduled using the external scheduler API, you should assign your communication user not only to SAP\_COM\_0064 but also to SAP\_COM\_0326. For more information on this issue, see [3358110](#).

## Examples

### Request

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobStatusGet?  
JobName='JobName'&JobRunCount='JobCount'
```

### Response

```
{  
  "d": {  
    "results": [  
      {  
        "JobName": "string"  
        "JobRunCount": "string"  
        "JobStatus": "char"  
        "ReturnCode": "integer" (0 = ok, <> 0 = error)  
      }  
    ]  
  }  
}
```

## 6.3.2.4.9.1.5 Abort a Running Application Job

Abort a running application job using the HTTP method POST.

## Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobRunCount	Mandatory	This is the ID of an application job.
JobName	Mandatory	This is the name of an application job.

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobAbort?  
JobName='JobName'&JobRunCount='JobCount'
```

### Response

```
{  
  "d": {  
    "results": [  
      {  
        "ReturnCode": "integer" (0 = ok, > 0 = error)  
      }  
    ]  
  }  
}
```

## 6.3.2.4.9.1.6 Cancel an Application Job

Cancel an application job in general using the HTTP method POST.

This operation is more general than JobAbort. JobCancel treats a job depending on its status:

- If a job is already running, JobCancel will cancel the job like JobAbort.
- If a job has not yet started, JobCancel will delete it.

## Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobRunCount	Mandatory	This is the ID of an application job.

Property	Necessity	Comment
JobName	Mandatory	This is the name of an application job.

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobCancel?
JobName='JobName'&JobRunCount='JobCount'
```

### Response

```
{
  "d": {
    "results": [
      {
        "ReturnCode": "integer" (0 = ok, > 0 = error)
      }
    ]
  }
}
```

## 6.3.2.4.9.1.7 Read Detailed Job and Step Information

Read detailed job and step information using the HTTP method GET.

This operation returns one line for each job step, where each line contains some global information of the job (like status, start time, end time) and step specific information like step status and step start time.

### Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobRunCount	Mandatory	This is the ID of an application job.
JobName	Mandatory	This is the name of an application job.

## Examples

### Request

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobinfoGet?  
JobName='JobName'&JobRunCount='JobCount'
```

### Response

```
{  
  "d": {  
    "results": [  
      {  
        "JobName": "string",  
        "JobRunCount": "string",  
        "JobStatus": "string",  
        "JobSdlDateTime": "DateTime",  
        "JobStartTime": "DateTime",  
        "JobEndDateTime": "DateTime",  
        "JobAppRC": "Integer",  
        "StepCount": "DateTime",  
        "StepStatus": "String",  
        "StepStartTime": "DateTime",  
        "StepAppRC": "Integer",  
      }  
    ]  
  }  
}
```

## 6.3.2.4.9.1.8 Read Application Job Status Texts

Get a list of all possible application job statuses and their corresponding status texts using the HTTP method GET.

### Request

Use the URL `<host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobStatusInfoSet` without parameters.

### Response

The operation returns a list of all application job statuses and their texts.

## Examples

### Request

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobStatusInfoSet
```

optional with language settings:

```
GET <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobStatusInfoSet?sap-language=EN
```

### Response

```
<d:JobStatus>C</d:JobStatus>
<d:JobStatusText>Canceled</d:JobStatusText>
```

or

```
<d:JobStatus>S</d:JobStatus>
<d:JobStatusText>Scheduled</d:JobStatusText>
```

and so on.

## 6.3.2.4.9.1.9 Restart a Failed Application Job

Create a copy of the failed application job and start it from or after the failed step using HTTP method **POST**.

### Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobRunCount	Mandatory	This is the ID of an application job
JobName	Mandatory	This is the name of an application job
JobRestartMode	Mandatory	This is the type of restart you can trigger

There are two restart modes:

- JobRestartMode='E': Restart the job from the failed step. The job steps before the failed step will be skipped.
- JobRestartMode='A': Restart the job after the failed step. The failed step and the steps before will be skipped.

## Response

The operation restarts a failed application job.

## Examples

### Request

```
POST <host>/sap/opu/odata/sap/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobRestart?  
JobName='JobName'&JobRunCount='JobCount'&JobRestartMode='E'
```

### Response

```
<d:JobName>JobName</d:JobName>  
<d:JobRunCount>JobCount</d:JobRunCount>  
<d:JobStatus>R</d:JobStatus>  
<d:ReturnCode>0</d:ReturnCode>
```

## 6.3.2.4.9.1.10 Check Parameters for Scheduling a Job

Check the parameter string that you intend to schedule a job with. This call makes the same checks as the *Check* button contained within the *Application Jobs* app.

### Request

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobTemplateName	Mandatory	This is the name of the application job.
JobParameterValues	Mandatory	These are the parameter values of the application job. For more information, see <a href="#">3383044</a> .
JobUser	Optional	This is the user under whom the application job is supposed to run. This field is kept optional for compatibility reasons only. It's recommended to always provide the job user when making the call.

## Response

The operation returns:

- Parameter String: An overview of the supplied parameter string. (In some applications, the supplied parameters can be changed by the framework. In such cases, a string with the changed parameters is returned in the response).
- Successful Indicator: Returns `true` when the supplied parameters are allowed. Otherwise it will return `false`.
- Changed Indicator: Returns `true` if the parameters have been adjusted. Otherwise it will return `false`.
- Dynamic Properties: <Reserved>
- Error Messages: Shows errors or warnings regarding the supplied parameter string.

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobScheduleCheck?  
JobTemplateName='TemplateName'&JobParameterValues='ParameterValueString'&JobUser=  
'UserName'
```

### Response

```
{  
    "d": {  
        "results": [  
            {  
                "JobParameterValues": "ParameterValueString",  
                "SuccessfulInd": "boolean",  
                "ChangedInd": "boolean",  
                "DynamicProperties": "",  
                "ErrMessages": "ErrorMessagesString",  
            }  
        ]  
    }  
}
```

## 6.3.2.4.9.1.11 Retrieve Application Job Parameters

Retrieve the application job parameters as a JSON string. This string can directly be used for scheduling an application job.

### Request (As a String)

You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobName	Mandatory	This is the name of the application job
JobCount	Mandatory	This is an alphanumerical job count value

## Response

The operation returns:

- Parameter String: A JSON string containing the application job parameters.

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/JobParamValuesGet?
JobName='JobName'&JobCount='JobCount'
```

### Response

```
<d:ParameterValues>{JSONSTRING}</d:ParameterValues>
```

## Request (Standard)

Retrieve the application job parameters. You can include the following properties in the URL of the request:

Property	Necessity	Comment
JobName	Mandatory	This is the name of the application job
JobCount	Mandatory	This is an alphanumerical job count value

## Response

The operation returns:

- Step Number: The number of the step where the parameter exists
- Parameter Name: The name of the parameter
- Sign/Option/Low/High: Values according to ABAP ranges

## Examples

### Request

```
POST <host>/sap/opu/odata/SAP/BC_EXT_APPJOB_MANAGEMENT;v=0002/
JobParamValuesStructGet?JobName='JobName'&JobCount='JobCount'
```

### Response

```
{
  "d": {
    "results": [
      {
        "StepNr": StepNumber,
        "JobParameterName": "ParameterName",
        "Sign": "I",
        "Option": "EQ",
        "Low": "LowValue",
        "High": "HighValue"
      }
    ]
  }
}
```

## 6.3.2.4.9.2 SAP\_COM\_0326

This communication scenario concerns application job scheduling authorizations.

### Context

If you create a communication arrangement based on SAP\_COM\_0326, your communication user will have the authorization to schedule any application job template under any business user. Furthermore, you'll also have the authorization to run all available operations on application jobs that were created by other users. This means that you can delete or cancel any application job created by any business user.

### Process Steps

1. Open the [Display Communication Scenarios](#) app.
2. Search for the scenario ID `SAP_COM_0326` and click on it.
3. Select [Create Communication Arrangement](#) in the top right corner.
4. Choose a name for your communication arrangement and click [Create](#).
5. You'll now need to create a communication system. To do this, click [New](#) next to the field [Communication System](#). For more information on general concepts of communication management, see [Communication Management \[page 2591\]](#).
6. Choose a new [System ID](#) and click [Create](#).

#### ⓘ Note

The *System Name* is filled automatically with the same name as the ID. We recommend leaving this as it is to prevent confusion.

7. In the *General* section, mark the *Inbound Only* checkbox.
8. Now, scroll down to *Users for Inbound Communication* and click "+" to add a new communication user.
9. Select *New User*. Enter a user name, a description, and a new password or select *Propose Password*. Click *Create* to save the user, then click *OK*.

#### ⓘ Note

Please don't use the *Certificate* section. At the moment, only basic authentication has been implemented.

10. Select *Save* to save your communication system.
11. You'll be forwarded to your communication arrangement. In the *Additional Properties* section, enter an *External Scheduler Company* and an *External Scheduler Product*.
12. Finally, select *Save* to save your communication arrangement.

### 6.3.2.4.10 Integrating Factory Calendar

Learn how to integrate an ABAP Platform system with the factory calendar.

#### Context

This communication scenario allows you to set up a connection to the import factory calendar customizing from another ABAP client. There can be at most one source client for an ABAP tenant.

#### Prerequisites

You have to connect your ABAP system via RFC. To do this, follow the steps described in [3148310](#) or install the required support package. The following RFC-enabled function modules are called during import:

- SCAL\_TO\_FHC\_MIG\_COMPLETED
- SCAL\_READ\_CUSTOMIZING
- FHC\_READ\_CUSTOMIZING

#### Procedure

1. Create a communication user in the *Maintain Communication Users* app. For more information, see [How to Create Communication Users \[page 2594\]](#).

2. Create a communication system in the *Communication Systems* app. For more information, see [How to Create Communication Systems \[page 2599\]](#).
3. Create a communication arrangement in the *Communication Arrangements* app. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

After creating the communication arrangement, assign the business catalog SAP\_CA\_BC\_IC\_LND\_CAL\_CA0\_PC or SAP\_CA\_BC\_IC\_LND\_CAL\_CA1\_PC to schedule an application job. The application job is required to pull the data of the three related IMG activities via the connection. The check logic of the application job and the transaction prevent the scheduling and execution in clients where the selected customizing objects are not editable.

#### ⓘ Note

Existing factory calendar customizing will be overwritten by the import. In case the automatic pull destination is configured, a warning is issued in the customizing maintenance UIs to prevent concurrent changes.

All changes are recorded on a task.

The source of the factory calendar depends on the migration state of the source client:

- If the source client has not yet been migrated, the cross-client calendar will be pulled
- If the source client has been migrated, the client-dependent calendar will be pulled

### 6.3.2.4.11 Integrating Outbound Emails Using SMTP

Learn how to integrate outbound emails using the Simple Message Transfer Protocol (SMTP).

#### Procedure

You must define the following configurations in a communication system. See [How to Create Communication Systems](#) for more information.

- Host
- Port
- Enable *Cloud Connector* to access your customer owned SMTP server
- Disable *Cloud Connector* when the communication is done directly to a publicly available SMTP server
- *SCC Location ID* of the SAP BTP Cloud Connector (only needed when the Cloud connector is enabled)

#### ⓘ Note

This parameter is only required if several SAP BTP Cloud connectors are used in one subaccount (to define the target SAP BTP Cloud Connector with the same *SCC Location ID*).

- A referenced outbound communication user with authentication method *User and Password*

### Note

- Instead of maintaining the information directly in the communication system, it's also possible to enter a referenced destination of type [MAIL](#) in the destination service. For more information, see [Create Mail Destinations](#).

Afterwards, create a communication arrangement for the `SAP_COM_0548` communication scenario using this communication system.

## Related Information

[Communication Systems \[page 2598\]](#)

[How to Create a Communication Arrangement \[page 2596\]](#)

[Integrating On-Premise Systems \[page 1184\]](#)

[Configure Access Control \(TCP\)](#)

## 6.3.2.4.12 Integrating SAP Enable Now Manager

The ABAP environment supports the extended content scenario of SAP Enable Now that allows you to add an additional source for help content to the SAP Companion plugin. You can edit the context-sensitive in-app help delivered by SAP and adapt it to your needs or include your own content.

To use the SAP Enable Now Manager, you have to create a communication arrangement that uses communication scenario `SAP_COM_0011`.

In addition, you have to install the SAP Enable Now Manager.

If you need further information, see the [SAP Companion Integration guide](#). The specifics for the ABAP environment are described in section *Set Up of SAP Companion for SAP S/4HANA Cloud and Similar Systems*.

## Related Information

[SAP Enable Now](#)

[SAP Companion Authoring](#)

## 6.3.2.4.13 Integrating UI Theme Designer

To set up the integration between SAP BTP ABAP Environment and UI theme designer, you can use the communication scenario `SAP_COM_0623`.

You can use `SAP_COM_0623` to establish a connection to the Cloud Foundry environment.

Create a communication arrangement in SAP BTP ABAP Environment to establish a connection to Cloud Foundry for fetching custom themes which can then be used in SAP BTP ABAP.

## Prerequisites

Your user needs a business role with the business catalog *Communication Management* (SAP\_CORE\_BC\_COM) as well as *Security* (SAP\_CORE\_BC\_SEC) assigned (e.g., the administrator user).

In Cloud Foundry, you need to have the environment for the UI theme designer prepared as described in [Portal Scenario](#).

A custom theme needs to be created and published in UI theme designer to be available for consumption in SAP BTP ABAP Environment. For more information, see [Create Themes — End to End Flow](#). The theme ID shown in UI theme designer will be used to define default theme or additional themes to be used in SAP BTP ABAP Environment.

An entitlement for the UI theme designer (theming) service is required in your global account and needs to be assigned to the subaccount.

## Creating an Instance for UI Theme Designer Service in Cloud Foundry

To consume published themes in SAP BTP ABAP Environment, an instance of UI theme designer service needs to be created in Cloud Foundry.

1. Navigate to the [Service Marketplace](#) in the Cloud Foundry subaccount. Select service *UI Theme Designer* (name: theming) and click *Create*.
2. In the creation dialog, enter basic info for the service instance
  1. Service: UI Theme Designer
  2. Plan: standard
  3. Runtime Environment: Cloud Foundry
  4. Space
  5. Instance Name
3. In the next screen, no further parameters need to be configured.
4. Review and verify the instance details in the following screen and confirm the creation with *Create*.
5. Make sure that a new instance for the *UI Theme Designer (theming)* service is successfully created.

## Creating a Service Key in Cloud Foundry

The simplest way to enable communication between SAP BTP ABAP Environment and Cloud Foundry is to create a service key in Cloud Foundry and use its content for configuring the communication arrangement.

1. Navigate to the space of your Cloud Foundry subaccount. Go to and select the instance for the UI theme designer.

- Now you can create a service key. For more information, see [Creating Service Keys \[page 255\]](#).
- Open the service key and copy its content (e.g., by clicking [Copy JSON](#)).

## Using mTLS for the Communication

To enable mTLS for the communication between SAP BTP ABAP Environment and the UI theme designer in Cloud Foundry, the steps are a little bit different:

- In SAP BTP ABAP Environment, go to [Maintain Client Certificates](#).
- Export the Client Default certificate. Choose one where [Valid to](#) is a date in the future.
- In Cloud Foundation, navigate to the space of your subaccount. Go to [Services](#) [Instances](#) and select the instance for the UI theme designer.
- Now create a service key with mTLS information. For more information, see section [Creating a Service Key in Cloud Foundry](#). When doing this via the Cockpit, enter the following text into the parameters field in the [New Service Key](#) wizard. Replace the `xxxxxxxxxxxxxxxxxxxxxx` with the text you find in your exported certificate.

### Sample Code

```
{
  "xsuaa": {
    "credential-type": "x509",
    "x509": {
      "certificate": "-----BEGIN CERTIFICATE----- xxxxxxxxxxxxxxxxxxxxxxxxx
-----END CERTIFICATE-----",
      "ensure-uniqueness": false,
      "certificate-pinning": true
    }
  }
}
```

- After creating the service key, open it and copy its content (e.g. by clicking on [Copy JSON](#)).

If the client certificate is not valid anymore, you need to repeat the described steps.

- As there can be only one Service Key at the same time for a Service Instance with a client certificate, you either need to delete the old one and create a new one, or you use two Service Instances where one is used active and the other can be used to update the Service Key.
- Download the new Client Default certificate as described above in steps 1 and 2.
- Recreate the Service Key as described above in steps 3 and 4. Copy this new key.
- Then go to the SAP BTP ABAP Environment, and in the launchpad choose [Communication Management](#) [Communication Arrangements](#).
- Select the previously created Communication Arrangement for `SAP_COM_0623`. Choose [Update by Service Key](#), paste the newly created Key and then choose [Update](#).

## Creating a Communication Arrangement in SAP BTP ABAP Environment

- Log on to the SAP BTP ABAP Environment system as administrator (or as a user with the necessary role/catalog assigned).

2. In the launchpad, navigate to the group *Communication Management* and choose the tile *Communication Arrangements*.
3. In the *Communication Arrangements* dialog, choose *New*.
4. Select communication scenario SAP\_COM\_0623 (*UI Theme Designer Integration*). If you want, change the proposed arrangement name.
5. Paste the service key into the field *Service Key*.
6. In the *Additional Properties* section, enter the custom theme as a default theme. (You can do or change this later in the detail screen).
7. Now click *Create*. With the help of the service key, everything is created in the background - communication system and communication arrangement - and all fields relevant for enabling a connection to Cloud Foundry are filled.

#### ⓘ Note

In a SAP BTP ABAP Environment system, there is only one communication arrangement for SAP\_COM\_0623 (connection to the Cloud Foundry environment) per client possible. If you have already an existing connection for custom themes and want to create a new connection, you must delete the old connection first.

8. Now you can maintain the fields for the communication arrangement. Besides the default theme ID, you can also maintain the field *Additional Theme IDs*. Here you can provide additional themes (resp. their IDs) from which the users can select one in the SAP Fiori launchpad personalization.

#### → Tip

We strongly recommend that you have a test phase first and enter the custom theme only as an additional theme. If tests by selected users are successful, you can enter your theme as the default theme (which applies to all users as soon as you save the communication arrangement).

9. Click on *Save*.

## Maintain Content Security Policy in SAP BTP ABAP Environment

To be able to correctly display custom themes in SAP BTP ABAP Environment, the runtime endpoint of UI theme designer (theming) service needs to be maintained as trusted site.

1. Log on to the SAP BTP ABAP Environment system as administrator (or as a user with the necessary role/catalog assigned).
2. In the launchpad, navigate to the group *Security* and choose the tile *Manage Content Security Policy*.
3. In the app, select the *Trusted Sites* tab and navigate into the *UI\_RESOURCES\_FONTS* allow list.
4. Press *Add*, then press *New* in the *Managed by Customer* section.
5. Fill in the theming runtime URL for your landscape and press *Save* (e.g. <https://theming-runtime.cfapps.eu10.hana.ondemand.com>)

#### → Tip

The theming runtime URL can be retrieved from the service key of the UI theme designer service instance, where it is the value of the "endpoints" → "runtime" property.

## Results

After a refresh in the FLP of SAP BTP ABAP Environment, your custom theme is available according to your configuration, as follows:

- If you have entered the custom theme as an *additional theme*, the SAP Fiori launchpad is unchanged but the theme is now available in the *User Actions* menu under ► *Settings* ► *Appearance* ▶.
- If you have set the custom theme as a *default theme* (only after thorough tests!), the SAP Fiori launchpad is displayed with the new theme for all users.

**Exception:** Users who have switched their theme previously in the user settings of the SAP Fiori launchpad keep their theme. But they can switch to the new default theme in their user settings since it is listed there.

### ⓘ Note

If there were changes within your instance on Cloud Foundry and you need to update the service key, you can copy the new content of the service key and use *Update by Service Key* within the communication arrangement.

## Troubleshooting

- **Theming runtime not included in Content Security Policy**

Custom themes are largely shown correctly, but many icons are not displayed.

Icons are loaded via fonts. This is reflected in the Developer Tools console (e.g. F12 in Google Chrome) as an error of the type "Refused to load the font ... because it violates the following Content Security Policy directive ..."

**Solution:** Theming runtime URL needs to be included into the *UI\_RESOURCE\_FONTS* allow list in *Manage Content Security Policy* app.

- **Custom Theme in Theming Service is not available and FLP in ABAP Environment is broken**

FLP in SAP BTP ABAP Environment is not usable because default theme or theme selected for business user cannot be loaded.

**Solution:** Is it possible to specify theme ID to be used for FLP in SAP BTP ABAP Environment using an URL parameter `sap-theme=<theme-id>`

See *Usage of the sap-theme URL parameter* in SAP Note [2043817](#).

This is especially useful if the FLP is broken, and the business user needs to select another theme to resolve the issue e.g., while using SAP-provided theme `sap-theme=sap_fiori_3`.

## 6.3.2.4.14 Integrating SAP Build Process Automation

You can integrate your ABAP environment with SAP Build Process Automation.

### 6.3.2.4.14.1 Configuring the SAP Build Process Automation Integration

You can enable the communication between the ABAP environment and SAP Build Process Automation.

This scenario enables ABAP environment applications to extend their business workflows or business processes with workflow capability of SAP Build Process Automation. It allows the provisioning of an API in the ABAP environment to start and control workflow instances running on workflow capability.

You set up the scenario using the following tasks:

1. [Create a Communication Arrangement Using a Service Key \[page 2909\]](#) (recommended)
2. [Create Destinations \[page 2910\]](#)
3. [AIF Monitoring \[page 2917\]](#)

#### 6.3.2.4.14.1.1 Create a Communication Arrangement Using a Service Key

To connect the ABAP environment and SAP Build Process Automation, you need a communication arrangement.

#### Procedure

1. Get the service key.
  - a. In the SAP BTP cockpit, navigate to your space. See [Navigate to Orgs and Spaces](#).
  - b. From the navigation pane, choose Services Service Instances .
  - c. Click the name of your SAP Build Process Automation instance.
  - d. From the navigation pane, choose *Service Keys*.
  - e. Select one entry, and copy the service key.
2. Open the *Communication Arrangements* app in your SAP S/4HANA Cloud system.
3. Create a communication arrangement.
4. Choose scenario *SAP\_COM\_0863*.
5. Enter a name for the arrangement.
6. Choose *Additional Properties* and enter:
  - Consumer Type (Workflow) - This is the default.
  - Optional. Consumer Type (Visibility)

- Choose the types relevant for SAP applications according to their documentation.
- Others: Leave empty.
7. Select a communication user (of which you know the password) or create a new one to use for inbound communication (from SAP BTP to your ABAP environment).
  8. Paste the service key into the corresponding field, and choose *Create*.

The password of the newly created user for the inbound communication is shown in an information message in the status bar. You need the user and password for maintaining a destination in SAP BTP.

## Results

You've created the communication arrangement, the communication system, and an inbound user. The consumer type is used to identify the communication arrangement. Therefore, you must only assign it to exactly one communication arrangement. You can assign multiple consumer types to one communication arrangement.

Consumer types can be added or removed later from the communication arrangement, if needed.

### 6.3.2.4.14.1.2 Create Destinations

In the SAP BTP cockpit, you need to create destinations to enable the communication to the ABAP environment. In addition, the destination must be enabled for use in SAP Build.

#### Procedure

1. In the SAP BTP cockpit, navigate to your subaccount. See [Navigate to Orgs and Spaces](#).
2. Choose *Connectivity* *Destinations*.
3. Choose *New Destination* *Blank Template*, and enter the following data:

Process Visibility Capability

Field	Value
Name	Enter the destination name.
Type	Select <i>HTTP</i> .
Description	Enter a description.

Field	Value
URL	API-URL of the ABAP environment. The format is: <code>https://*****.ondemand.com/</code> . Set the destination URL to the API endpoint URL found in the service key that was created for the service instance of SAP Build Process Automation and which was used creating the communication arrangement. Access the communication arrangement and copy the API URL displayed under  <a href="#">Common Data</a>  .
Proxy Type	<i>Internet</i>
Authentication	Select <a href="#">Basic Authentication</a> .
User	(Inbound) communication user used in the communication arrangement
Password	Set the password to the secret. Password of the (inbound) communication user used in the communication arrangement.
Additional Properties	Add the following properties: <ul style="list-style-type: none"> <li>“sap.applicationdevelopment.actions.enabled” → true</li> <li>“sap.processautomation.enabled ” → true</li> </ul>

4. Save your entries.
5. Open the Settings of SAP Build on your subaccount.
6. Choose  [Destinations](#) .
7. Select the destination that you have created in the SAP BTP cockpit and click [Add](#).

## Results

You have created a destination in your SAP BTP subaccount that can be used for the callback to the SAP BTP ABAP environment. The callback indicates the completion of a process on SAP Build Process Automation.

## Related Information

[Configure SAP Build Process Automation Destinations](#)

### 6.3.2.4.14.1.3 Model Action Project to Notify the ABAP Environment About Workflow Completion

As the workflow capability within SAP Build Process Automation doesn't support the enterprise messaging service, you must model an additional action right before each end event for each process definition, that was

started from the ABAP environment. This action calls an inbound service and notifies the ABAP Environment that the process that has been started on SAP Build Process Automation is now completed.

## Prerequisites

You've modeled a project with a process within SAP Build.

## Context

The action triggers the completion in the ABAP environment tenant and must be called exactly once.

With this approach, cancellations on workflow instances directly from the workflow capability API aren't communicated to the ABAP environment tenant.

## Procedure

1. Create a new project in SAP Build by choosing  *Build an Automated Process* .
2. Upload the API specification.

### Sample Code

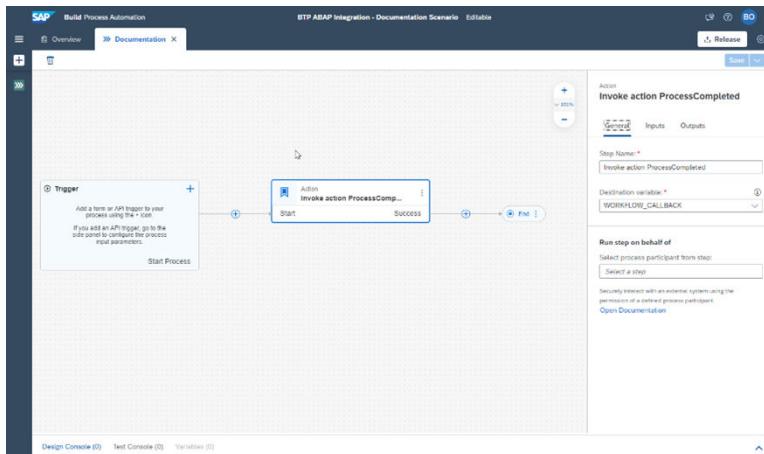
```
<?xml version="1.0" encoding="utf-8"?>
<edmx:Edmx Version="4.0" xmlns:edmx="http://docs.oasis-open.org/odata/ns/
edmx" xmlns="http://docs.oasis-open.org/odata/ns edm">
    <edmx:DataServices>
        <Schema
Namespace="com.sap.gateway.default.api_workflow_notification.v0001"
Alias="SAP_self">
            <ComplexType Name="CT_NOTIFICATION_RESULT">
                <Property Name="notificationProcessed" Type="Edm.Boolean"
Nullable="false"/>
            </ComplexType>
            <Action Name="ProcessCompleted">
                <Parameter Name="workflowInstanceId" Type="Edm.String"
Nullable="false" MaxLength="36"/>
                <Parameter Name="outcome" Type="Edm.String" Nullable="false"
MaxLength="255"/>
                <ReturnType
Type="com.sap.gateway.default.api_workflow_notification.v0001.CT_NOTIFICATION_RESULT"
Nullable="false"/>
            </Action>
            <EntityContainer Name="Container">
                <ActionImport Name="ProcessCompleted"
Action="com.sap.gateway.default.api_workflow_notification.v0001.ProcessCompleted"/>
            </EntityContainer>
        </Schema>
    </edmx:DataServices>
</edmx:Edmx>
```

3. Enter a project name.
4. In the *Add actions ...* dialog, choose only the service operation *POST /Process Completed* and then choose *Add*.
5. Modify the project settings as follows
  - a. Enable CSRF tokens for all actions in the project, and set *Token Fetch End Point* to „/“.
  - b. Set the URL prefix to „/sap/opu/odata4/sap/api\_workflow\_notification/default/sap/api\_workflow\_notification/0001“.
6. Save the project.
7. Release the project and publish it to the library.

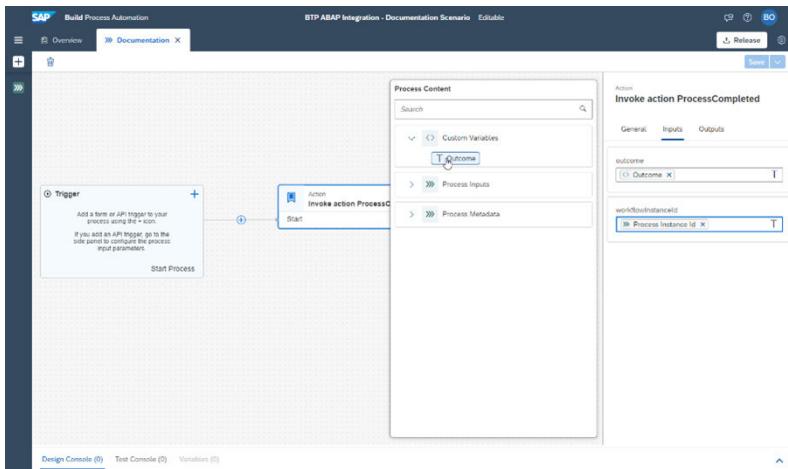
## Using the Action in a Process

### Procedure

1. Include a new action step as last step in your process by choosing .
2. Search for „Invoke action ProcessCompleted“ and choose *Add*.
3. Specify a destination variable in the action step details – create a new variable if necessary.



4. Configure the inputs of the action step.
  - a. From *Process Metadata*, assign *Process Instance Id* to the *workflowInstanceld* input.
  - b. Optional. To communicate the result of the process, assign any relevant variable to the *outcome* input.



- When deploying your project, specify the destination created in the SAP BTP cockpit for the destination variable used in the action step.

### 6.3.2.4.14.1.3.1 Using the Workflow Proxy API on SAP BTP ABAP Environment

To start and control a workflow process on SAP Build Process Automation from your application running in the SAP BTP ABAP environment, the following released APIs can be used.

Artifact	Description
CL_SWF_CPWF_API_FACTORY_A4C	Factory for instantiating an API instance
IF_SWF_CPWF_API	Actual API interface
I_CPFW_INST	RAP façade for the workflow API to be used within RAP behavior definitions

For more information, see the ABAP documentation of these objects in the system.

### 6.3.2.4.14.1.4 Modeling Extension Processes

As the SAP Build Process Automation processes don't integrate using SAP Event Mesh, you must ensure the communication back to SAP S/4HANA Cloud through the process models.

#### Prerequisites

- You've created the destination for the workflow capability integration. See [Create Destinations \[page 2910\]](#).

- You have the necessary roles to develop and deploy SAP Build Process Automation projects. See [Managing Project Members](#).

## Context

To ensure the communication, you have to model an additional service task right before each end event for each process, that was started from SAP S/4HANA Cloud.

### ⓘ Note

Cancellations of process instances on SAP Build Process Automation aren't communicated to the SAP S/4HANA Cloud system.

## Procedure

1. To create an action for communicating the process completion to the SAP S/4HANA Cloud system, create an actions project. See [Model Action Project to Notify the ABAP Environment About Workflow Completion \[page 2911\]](#).
2. Create a SAP Build Process Automation process.
  - a. Create a business project. See [Create a Business Process Project](#).
  - b. Create a data type for the input of the process.

The data type is for receiving the input from the SAP S/4HANA Cloud system. You assign it to the process trigger. Choosing New Field, you enter the following Names:

- **SystemID**
- **SAPObjectNodeType**
- **SAPObjectNodeTypeKey**
- **Result**

Add children to the **SAPObjectNodeTypeKey** field and name them **Key1** to **Key6**.

The data type then looks as follows:

Name	Type	Sample	List	Required	
SystemID	String		No	No	<button>New Child</button>
SAPObjectNodeType	String		No	No	<button>New Child</button>
✓ SAPObjectTypeKey	Object		No	No	<button>New Child</button>
Key1	String		No	No	<button>New Child</button>
Key2	String		No	No	<button>New Child</button>
Key3	String		No	No	<button>New Child</button>
Key4	String		No	No	<button>New Child</button>
Key5	String		No	No	<button>New Child</button>
Key6	String		No	No	<button>New Child</button>
Result	String		No	No	<button>New Child</button>

- c. Create a business process. See [Create a Business Process](#).
- d. Receive the input from the SAP S/4HANA Cloud system.
  - 1. Select the trigger tile.
  - 2. Open the *Inputs/Outputs* tab.
  - 3. Choose *Configure* next to *Process Inputs*.
  - 4. Choose add *Input*.
  - 5. Enter **ExchangeData** in the *Name* field, and select the previously created data type.
  - 6. Choose *Apply*.
- 3. Communicate the process completion to the SAP S/4HANA Cloud system.
  - a. Add the action that you created before each end event. For information about how to add actions, see [Add Actions to a Process](#).
  - b. On the *General* tab, create a destination variable for the completion callback so that you can map it to the destination on deployment.
  - c. On the *Inputs* tab, assign to the *workflowInstanceId* input of the action.
  - d. Optional: On the *Inputs* tab, assign any variable to the outcome input of the action to indicate the result of the extension process to the caller.
- 4. Release your project.
- 5. Deploy your project.
  - a. In the *Runtime Variables* step, assign the destination for the workflow capability integration to the destination variable that you created.

## Related Information

[Configure Service Tasks](#)

## 6.3.2.4.14.2 AIF Monitoring

The Application Interface Framework (AIF) enables you to monitor background communication with your connected SAP Build Process Automation tenant.

### Procedure

First, you need to assign business roles to a business user and then to assign recipients to this user.

## 6.3.2.4.14.2.1 Maintain Business Roles and Business Users

As a preparation to monitor messages for the integration of the SAP Build Process Automation, you assign business roles to a business user.

### Procedure

1. Log on to the SAP Fiori launchpad as an administrator user.
2. With the *Maintain Business Roles* app, create and edit business roles and add business catalogs to the roles.
  - a. To grant access to the *Assign Recipients to Users* app, add the business catalog *Communication Management – Message Monitoring Configuration* (SAP\_CA\_BC\_COM\_CONF\_PC) to a business role.
  - b. To grant access to the *Message Dashboard* app, add the business catalog *Business Network Integration* (SAP\_BR\_CONF\_EXPERT\_BUS\_NET\_INT) to a business role.

Both business catalogs are also contained in the business role template *Configuration Expert – Business Network Integration* (SAP\_BR\_CONF\_EXPERT\_BUS\_NET\_INT).

3. With the *Maintain Business User* app, provide business users with access rights.

## 6.3.2.4.14.2.2 Monitor Messages for SAP Build Process Automation Integration

You can monitor the background processes for the integration of the SAP Build Process Automation.

### Prerequisites

You have assigned business roles and business users.

## Context

With the [Message Dashboard](#) or [Message Monitoring Overview](#), you can monitor and administrate the SAP Build Process Automation background processing. You can manually trigger or cancel monitored background tasks that have issues.

AIF messages are deleted after a retention time, which can be configured in AIF customizing. The default value for all workflow-relevant namespaces and interfaces is set to 7 days.

## Procedure

1. Log on to the SAP Fiori launchpad as an administrator user.
2. With the [Assign Recipients to Users](#) app, assign the corresponding recipients to a business user.
  - a. Select the user for which you want to perform the assignment.
  - b. Repeatedly choose [Assign](#) and successively enter the following recipients:

Namespace	Recipient
<a href="#">/SWFCP</a> (for workflow capability)	<i>REC_PROC_BGRFC (process bgRFC)</i>
	<i>REC_RAISE_EVT (Recipient for raise of event)</i>
<a href="#">/SWFPV</a> (for process visibility capability)	<i>EVENT_PUBL (Process bgRFC)</i>

Also choose the message types that the user is allowed to view.

For more information, see [Assigning Users to Recipients](#).

3. To view the results, open the [Message Dashboard](#) app.
4. To view the triggered messages, use the [Calendar Monitor](#) to select the date range and choose [Search](#).
5. Alternatively, view the results with the [Message Monitoring Overview](#) app.

The overview displays the interfaces and message types to which you're subscribed. To see more details, click a message type.

## Related Information

[Blog on Working with the Message Dashboard](#) 

## 6.3.2.4.15 Integrating Workflow

You can enable the communication between the ABAP environment and the workflow capability within the SAP Workflow Management service.

This scenario enables the ABAP environment applications to extend their business processes by using the workflow capability in the Cloud Foundry environment. It allows the provisioning of an API in the ABAP environment to start and control workflow instances running on workflow capability.

You set up the scenario using the following tasks:

1. [Create a Communication Arrangement Using a Service Key \[page 2919\]](#) (recommended)
2. [Create a Destination \[page 2920\]](#)
3. [Maintain Business Roles and Business Users \[page 2924\]](#)

### 6.3.2.4.15.1 Create a Communication Arrangement Using a Service Key

To connect the ABAP environment and the SAP BTP cockpit, you need a communication arrangement.

#### Procedure

1. Get the service key.
  - a. In the SAP BTP cockpit, navigate to your space. See [Navigate to Orgs and Spaces](#).
  - b. From the navigation pane, choose .
  - c. Click the name of your workflow capability instance.
  - d. From the navigation pane, choose *Service Keys*.
  - e. Select one entry, and copy the service key.
2. Open the *Communication Arrangements* app in your ABAP environment.
3. Create a communication arrangement.
4. Choose scenario *SAP\_COM\_0542*.
5. Enter a name for the arrangement.
6. Select a communication user to use for inbound communication (from SAP Business Technology Platform to your ABAP environment).
7. Paste the service key into the corresponding field, and choose *Create*.

The password of the newly created user for the inbound communication is shown in an information message in the status bar. You need the user and password for the destination in the SAP Business Technology Platform.

## Results

You've created the communication arrangement, the communication system, and an inbound user. The communication arrangement is assigned to the DEFAULT consumer type. The consumer type is used to identify the communication arrangement. Therefore, you must only assign it to exactly one communication arrangement. You can assign one communication arrangement to multiple consumer types.

## Adjust the Consumer Type

### Procedure

1. Select the communication arrangement, and choose *Edit*.
2. Set the consumer type to DEFAULT.

## 6.3.2.4.15.2 Create a Destination

In the SAP BTP cockpit, you need to create a destination to enable the communication to the ABAP environment.

### Procedure

1. In the SAP BTP cockpit, navigate to your subaccount. See [Navigate to Orgs and Spaces](#).
2. Choose .
3. Choose [New Destination](#), and enter the following data:

Field	Value
Name	Enter the destination name. This name is also used in the SAP Web IDE as the destination name in the service task properties.
Type	Select <i>HTTP</i> .
Description	Enter a description.

Field	Value
URL	URL for the ABAP environment. The format is: <code>https://*****.ondemand.com/</code> . Set the destination URL to the authorization endpoint URL found in the service key that was created for the workflow capability when creating the communication arrangement. Access the communication arrangement and copy the root URL under  <a href="#">Inbound Services</a>  <a href="#">Service URL/Service Interface</a> .
Proxy Type	<i>Internet</i>
Authentication	Select <i>BasicAuthentication</i> .
User	Set the user to the client ID as specified for the inbound communication.
Password	Set the password to the secret.

- Save your entries.

### 6.3.2.4.15.3 Set Scopes for the Service Instance Using the Command Line Interface

To set the necessary scopes of a service instance, you need to install a new service interface with all scopes needed or you update an already existing service interface.

#### Procedure

- To update the workflow capability instance with the necessary scopes, save the following JSON file that specifies all necessary scopes:

```
{
  "authorities": [
    "WORKFLOW_INSTANCE_START",
    "WORKFLOW_DEFINITION_GET",
    "WORKFLOW_INSTANCE_GET",
    "WORKFLOW_INSTANCES_UPDATE",
    "WORKFLOW_INSTANCE_CANCEL",
    "WORKFLOW_INSTANCE_GET_ERROR_MESSAGES",
    "WORKFLOW_INSTANCE_GET_CONTEXT",
    "WORKFLOW_INSTANCE_GET_EXECUTION_LOGS",
    "MESSAGE_SEND",
    "TASK_GET"
  ]
}
```

To set these scopes, see *Update an existing service instance* in [Enable Technical Authentication](#).

- Model the workflow as described in [Developing Applications with Workflow Capability](#).

## 6.3.2.4.15.4 Model Service Task to Complete the Workflow in the ABAP Environment

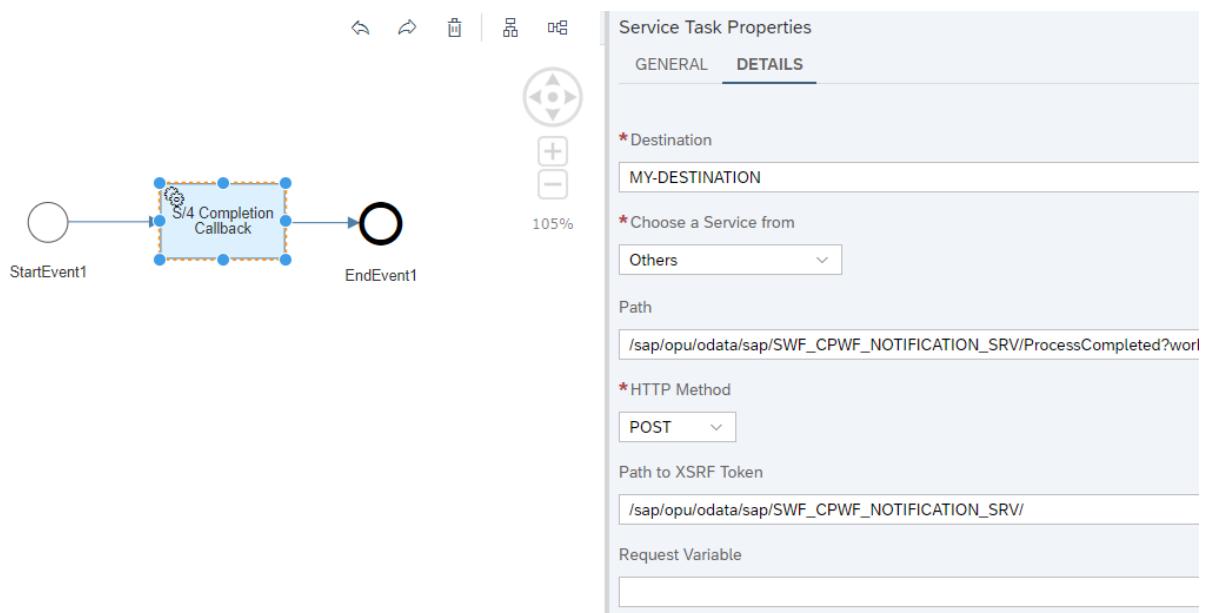
As the workflow capability doesn't support the enterprise messaging service, you must model an additional service task right before each end event for each workflow definition, that was started from the ABAP environment.

### Prerequisites

You have modeled a workflow. See [Modeling a Workflow](#).

### Context

This task triggers the completion in the ABAP environment tenant and must be called exactly once.



With this approach, cancellations on workflow instances directly from the workflow capability API aren't communicated to the ABAP environment tenant.

### Procedure

1. Open the workflow that you have modeled in the workflow editor.
2. Select the service task, and go to the **DETAILS** tab.

3. Enter the following data:

Field	Value
Destination	Enter the name of the destination you've created in Cloud Foundry.
Choose a Service from	<i>Others</i>
Path	/sap/opu/odata/sap/SWF_NOTIFICATION_SRV/ ProcessCompleted?workflowInstanceId='\\$ {info.workflowInstanceId}'
HTTP Method	<i>POST</i>
Path to XSRF Token	/sap/opu/odata/sap/SWF_NOTIFICATION_SRV/

### 6.3.2.4.15.5 AIF Monitoring

AIF (Application Interface Framework) enables you to monitor inbound and outbound messages in the ABAP environment and allows you to analyze communication issues.

#### Procedure

First, you need to assign business roles to a business user and then to assign recipients to this user.

#### Related Information

[Maintain Business Roles and Business Users \[page 2924\]](#)

[Monitor Messages for Workflow Capability Integration \[page 2924\]](#)

## 6.3.2.4.15.5.1 Maintain Business Roles and Business Users

As a preparation to monitor messages for background workflow processes, you create a business role and a business user.

### Procedure

1. Log on to the SAP Fiori launchpad as an administrator user.
2. With the *Maintain Business Roles* app, create and edit business roles and add business catalogs to the roles.
  - a. To grant access to the *Assign Recipients to Users* app, add the business catalog *Communication Management – Message Monitoring Configuration* (SAP\_C\_A\_BC\_COM\_CONF\_PC) to a business role.
  - b. To grant access to the *Message Dashboard* app, add the business catalog *Business Network Integration* (SAP\_C\_A\_BC\_COM\_ERR\_PC) to a business role.
3. With the *Maintain Business User* app, provide business users with access rights.

## 6.3.2.4.15.5.2 Monitor Messages for Workflow Capability Integration

You can monitor the background processes for workflow capability integration.

### Prerequisites

You have assigned business roles and business users.

### Context

With the *Message Dashboard* or *Message Monitoring Overview*, you can monitor and administrate the SAP Business Workflow background processing. You can manually trigger or cancel monitored background tasks that have issues.

AIF messages are deleted after a retention time, which can be configured in AIF customizing. The default value for all workflow-relevant namespaces and interfaces is set to 7 days.

## Procedure

1. Log on to the SAP Fiori launchpad as an administrator user.
2. With the [Assign Recipients to Users](#) app, assign the corresponding interface to a business user.
  - a. In the *Namespace* field, enter the namespace `/SWFCP` for the workflow capability integration.
  - b. Select the interfaces using the recipient names:
    - REC\_PROC\_BGRFC (Process bgRFC)
    - REC\_RAISE\_EVT (Recipient for raise of event)
  - c. Choose the message types the user is allowed to view:
    - Application Error or Technical Error
    - Info
    - Success
    - Warning
    - Application Error
    - Technical Error
    - None

For more information, see [Assigning Users to Recipients](#).

3. To view the results, open the [Message Dashboard](#) app.
4. To view the triggered messages, use the [Calendar Monitor](#) to select the date range and choose [Search](#).
5. Alternatively, view the results with the [Message Monitoring Overview](#) app.

The overview displays the interfaces and message types to which you're subscribed. To see more details, click a message type.

## Related Information

[How to enable users to work with the message dashboard](#) 

### 6.3.2.4.16 Integrating Business Event Logging

## Context

There are three integration scenarios supporting data flow from Business Event Logging to the target systems:

- SAP\_COM\_0A54: This integration scenario is for data federation/extraction from Business Event Logging to target systems.
- SAP\_COM\_0532: This integration scenario is for data replication from Business Event Logging to target systems. You can use the schema `API_BusEvtLogEvent` from SAP\_COM\_A054 communication scenario.

For more information on this communication scenario, see [Integrating SQL Services using SAP Datasphere](#).

### 6.3.2.4.16.1 Prerequisites

- Create a communication user in the Maintain Communication Users app. For more information, see [How to Create Communication Users \[page 2594\]](#).

#### ⓘ Note

Ensure that you use the same communication user for SAP\_COM\_A054 and SAP\_COM\_532 for replication scenario.

- Create a communication system in the Communication Systems app. For more information, see [How to Create Communication Systems \[page 2599\]](#).
- Create a communication arrangement in the Communication Arrangements app. For more information, see [How to Create a Communication Arrangement \[page 2596\]](#).

### 6.3.2.4.16.2 Consumption of Data

Data can be consumed using either External ODBC Clients or using SAP Datasphere:

#### **Data Consumption using External ODBC Clients**

For more information on data consumption, see Consumption of the SQL Service for Data Federation [Consumption of the SQL Service for Data Federation \[page 1227\]](#).

#### **Data Consumption using SAP Datasphere**

For more information on data consumption, see Consumption of the SQL Service for Data Replication [Consumption of the SQL Service for Data Replication \[page 1229\]](#).

## 6.3.3 Technical Operations

Learn more about what you can do to operate your ABAP environment.

As an administrator in the ABAP environment, you need metrics and alerts to monitor the availability and performance of your systems. You also need an indication whether your purchased service volume has been exhausted. Therefore, the following metrics and alerts are particularly relevant for the ABAP environment:

- Utilization metrics for computing, storage, and network resources
- Performance metrics
- Exceptional situations such as aborted jobs or runtime errors

In the ABAP environment, you have different tools available to support you with your tasks.

## Local Monitoring

For local monitoring of your ABAP environment, you can use the technical monitoring cockpit. In the technical monitoring cockpit, multiple monitoring screens help you identify and solve bottlenecks and performance issues by giving you an in-depth view on important technical data of the database and application server, such as CPU, memory, network, storage, performance, time spent, workload, and configuration. You can use the technical monitoring cockpit also for a profound analysis of the most important entities, such as work processes, tables, SQL traces, and SQL statements.

For more information, see [Technical Monitoring Cockpit \(Cloud Version\)](#).

## System Landscape Monitoring

For monitoring an entire system landscape, consider using a central monitoring and alerting infrastructure. If you use SAP Focused Run or SAP Cloud ALM for monitoring your system landscape, you can use the integrated health monitoring for the ABAP environment.

For more information, see [Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#).

### 6.3.3.1 Common Tasks: Quick Reference

Learn how you can perform typical tasks during technical operations using the available tools in the ABAP environment.

#### ⓘ Note

To be able to use the available local apps for technical monitoring, you need a business role based on the business role template `SAP_BR_ADMINISTRATOR` or `SAP_BR_DEVELOPER`.

#### 6.3.3.1.1 Analyzing the ABAP Resource Utilization of the ABAP Environment

### Context

When you're planning, for example, system capacities or facing performance issues with SAP or partner cloud services running on the ABAP environment, you want to understand the resource utilization of your ABAP environment.

## Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for [Application System Overview](#).  
The technical monitoring cockpit opens.
2. To gain insights into the memory utilization of the ABAP application server, choose the [Memory Utilization](#) tile.
3. To gain further insights into the work process utilization of the ABAP application server, choose the [Sampled Work Process Data: Backgr. Work Process Utilization](#) and the [Sampled Work Process Data: Dialog Work Process Utilization](#) tiles.  
A high load on the background work processes isn't critical if it's only for a short duration. If the background work process utilization is at 100%, jobs are started delayed.  
For more information about the work process utilization screens, see [Sampled Work Process Data: Background Work Process Utilization](#) and [Sampled Work Process Data: Dialog Work Process Utilization](#).
4. To gain more insights into the application workload of the system, choose the [System Workload](#) tile from the application system dashboard.  
Click on any peak in the chart to get the top 10 most contributing ABAP requests in the [System Workload: Details](#) section.



For more information about the *System Workload* screen, see [System Workload](#).

### 6.3.3.1.2 Analyzing the System Workload

You want to know more about the ABAP system workload.

#### Context

Using the *System Workload* screen, you benefit from a seamless connection between an overview of your system workload and single ABAP statistics records (main records and subrecords) down to SQL statements

and their prepared plan. This gives you an excellent starting point for further performance analysis on SAP HANA level in the case of high database response times, which is true especially in the example below. Of course, you might also find records with a high ABAP CPU time where you don't need to further investigate the execution of SQL statements. In that case, a performance analysis on SAP HANA level would not be needed.

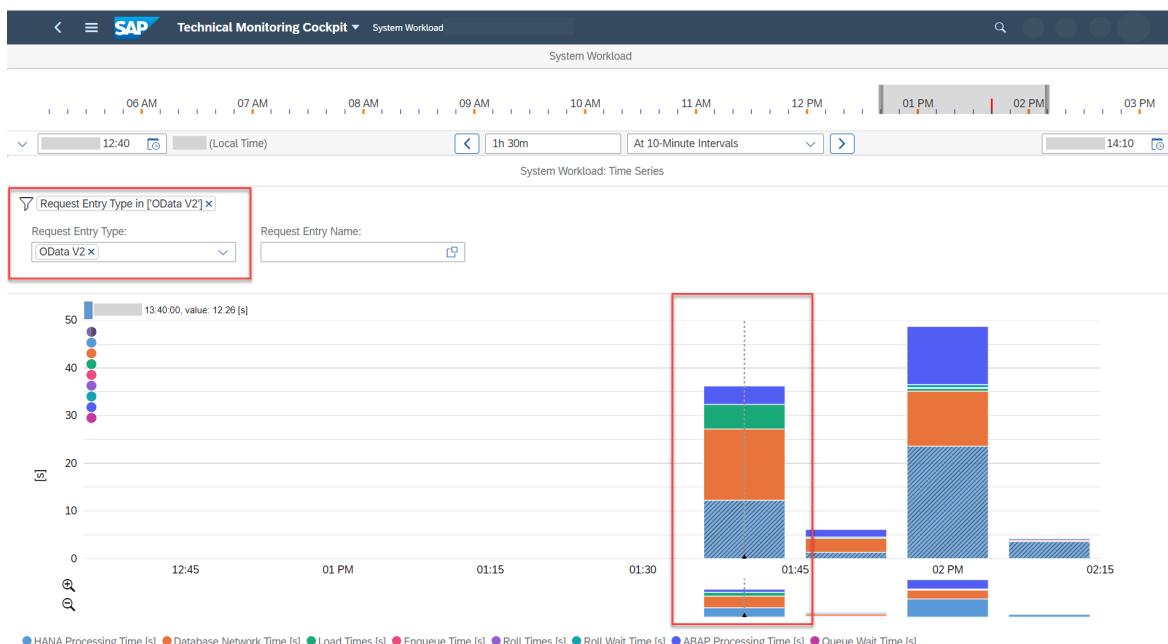
In the following example, let's analyze the usage of an application built according to the ABAP RESTful Application Programming Model (RAP). We're going to use the well-known demo application for flight booking, which is often used as a reference scenario in SAP contexts.

## Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for [System Workload](#).

The technical monitoring cockpit opens with the [System Workload](#) screen.

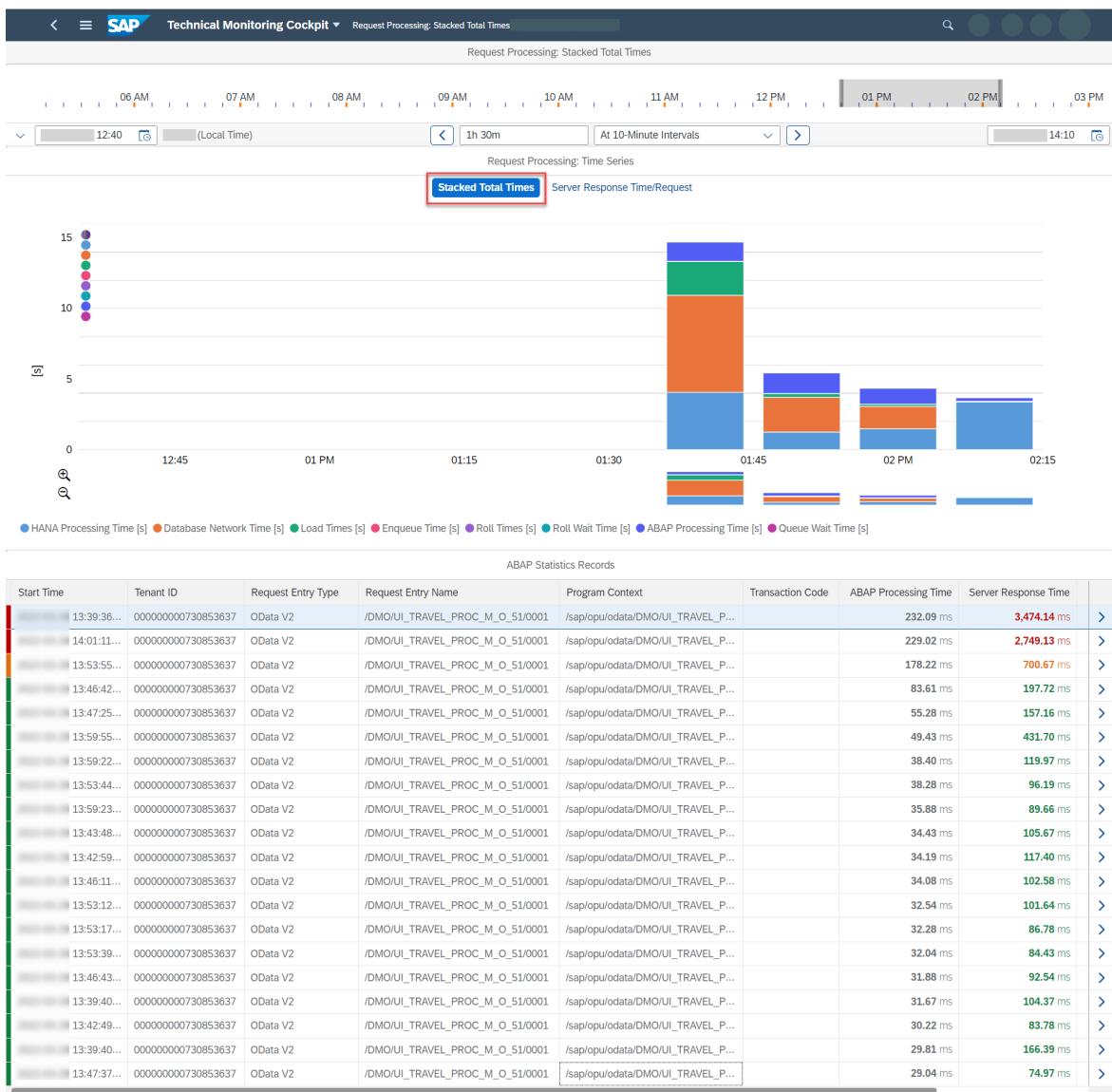
2. Let's assume you're interested in the behavior of the OData V2 services and set a filter on [Request Entry Type](#) for [OData V2](#).
3. In the chart, click on the HANA processing time that seems high, for example, at 01:40 p.m.



The [System Workload: Details](#) section below the chart changes accordingly and now displays the top contributors to the HANA processing time at the time you clicked in the chart. The contributors are grouped by request entry type and request entry name.

System Workload: Details								
Request Entry Type	Request Entry Name	Program Context	Transaction Code	HANA Processing Time	Avg. Server Response Time			
OData V2	/DMO/UI_TRAVEL_PROC_M_O_51/0001	/sap/opu/odata/DMO/UI_TRAVEL_PROC_M_O_51/\$batch		4.07 s	0.65 s	>		
OData V2	/DMO/UI_TRAVEL_PROC_M_O_51/0001	/sap/opu/odata/DMO/UI_TRAVEL_PROC_M_O_51/\$metadata			3.41 s	2.56 s	>	
						>		
						>		
						>		
						>		
						>		
R					0.01 s	0.07 s	>	
T					12.26 s	0.96 s	>	

4. To get more details, let's choose that first highlighted entry in the list. The subsequent *Request Processing* screen shows the behavior of this specific workload (the flight processing service) over time:

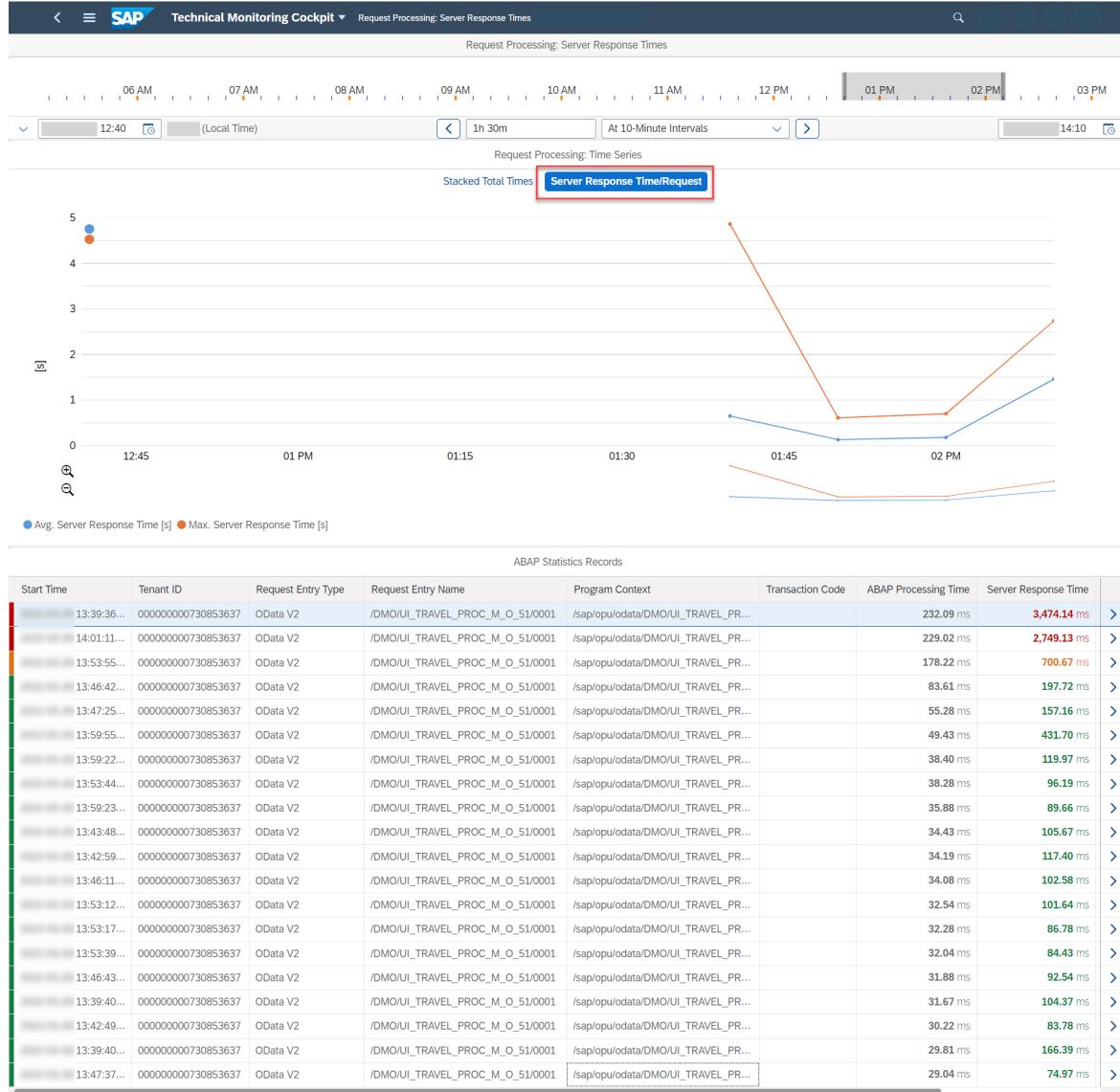


The chart shows the stacked total times of the metrics, and - depending on what you click in the chart - the table below shows the relevant detail view, that is, a list of a maximum of 20 single ABAP statistics records

for this workload sorted by the metric you chose in the chart. This is very helpful for your further analysis: You're coming from a workload perspective and now get the single records the workload consists of.

The color coding of the single records is based on their deviation from the average server response times (top to bottom): Red for very high, orange for above normal, and green for normal server response times.

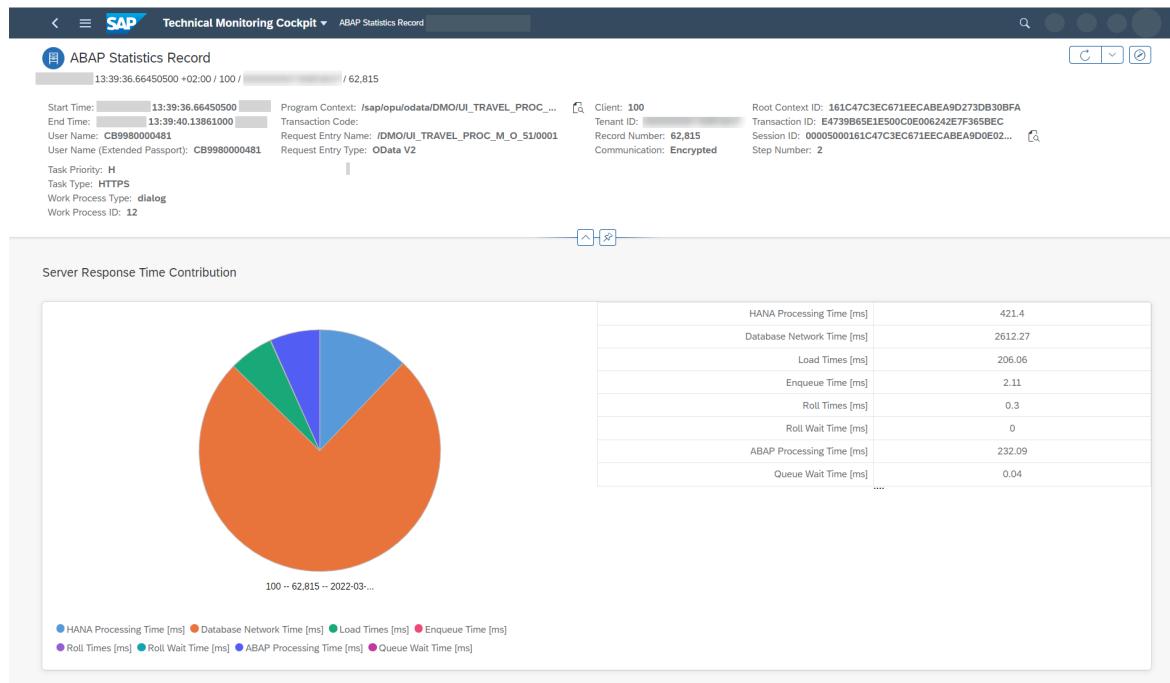
- Let's switch views by choosing the *Server Response Time/Request* button:



Like in the *Stacked Total Times* view, the color rating of the single records is based on their deviation from the average server response times. The average server response time per request is around one second here. However, there are some requests with higher times, so let's have a closer look at a "bad" request.

- In the table, choose the first red entry (the one with 3,474.14 ms).

On the following *ABAP Statistics Record* screen, you get more detailed information about this record. The pie chart for the server response time contributions clearly shows that the response time is dominated by the database network time:



What's also noticeable is the DEFAULT database connection that apparently fired 1,284 database requests:

Database Connections						
Connection Name	DB Requests ...	Database Processing Time	HANA Processing Time	HANA CPU Time	HANA Maximum Memory	
DEFAULT	1,284	2,969.87 ms	418.35 ms	393.61 ms	1,206.57 kB	>
	1	25.31 ms	0.00 ms	0.00 ms	0.00 kB	>
	7	29.48 ms	3.04 ms	1.72 ms	35.41 kB	>
	2	9.00 ms	0.00 ms	1.85 ms	17.37 kB	>

7. Let's dig deeper: In the *Database Connections* section of the *ABAP Staticstics Record* screen, choose the first entry (connection name DEFAULT) to get more information.

The *Database Connection Details* screen opens and shows that 1,284 logical database requests were fired to fetch almost 13,000 database rows, which explains the high network time: It was caused by the many database requests.

Database Request	Request Time	Logical Database Requests	Physical Database Requests	Database Rows	Buffer Rows	Request Time/Database Row
Direct Read	0.00 ms	38	1,159	0	31	0.00 ms
Delete	2.91 ms	2	2	0	0	0.00 ms
<b>Totals</b>	<b>2,963.58 ms</b>	<b>1,284</b>	<b>1,164</b>	<b>12,905</b>	<b>676</b>	<b>0.15 ms</b>
Insert	4.68 ms	2	2	3	0	1.56 ms
Update	5.98 ms	1	1	1	0	5.98 ms
Sequential ...	2,021.24 ms	1,241	0	12,901	645	0.15 ms
Other	928.76 ms	0	0	0	0	0.00 ms
			....			

8. Another thing you may want to check in this example is the *SQL Statements* section on the *ABAP Statistics Record* screen: The table shows the top SQL statements contributing to the HANA processing time.

### ⓘ Note

The SQL statements are available in the single record by default. You do **not** need to activate any tracing explicitly.

Let's see if the access on CDS views is OK (in this case on the booking view). Place the cursor on the entry so that it's highlighted and simply choose *Analyze Statement* to resolve the SQL statement:

Connection Name	Statement	Analyze Statement	Database Processing Time	HANA Processing Time	HANA CPU Time	HANA Max. Memory
DEFAULT	SELECT "LDATA" FROM "REPOLOAD" WHERE "PROGNAME" = ? AND "R3STATE" = ? AND "MACH" = ? AND "MAJO...		1,298.39 ms	107.56 ms	68.99 ms	47.57 kB
DEFAULT	SELECT "UNAM", "UDAT", "UTIME", "L_DATALG", "Q_DATALG", "SDAT", "STIME", "MINOR_VERS", "MAJOR_V...		956.39 ms	157.92 ms	107.72 ms	31.88 kB
DEFAULT	SELECT "QDATA" FROM "REPOLOAD" WHERE "PROGNAME" = ? AND "R3STATE" = ? AND "MACH" = ? AND "MAJ...		98.66 ms	8.16 ms	5.01 ms	43.55 kB
DEFAULT	SELECT /* Contains Native SQL */ "BOOKINGDATE", "BOOKINGID", "BOOKINGID" "BOOKINGID", "BOOKINGST...	Analyze Statement	78.15 ms	75.44 ms	75.77 ms	158.71 kB
DEFAULT	SELECT /* FDA READ */ "DDLANAME", "DEFINITION_KEY", "KEY_GUID", "VALUE" FROM "DDLA_RT_ANNO\$..."		64.26 ms	0.85 ms	0.25 ms	6.54 kB
DEFAULT	SELECT /* FDA WRITE */ "PERS", "TRAVEL_ID", "PERS", "BOOKING_ID", "PERS", "BOOKING_SUPPLEMENT_ID..."		30.02 ms	27.35 ms	26.97 ms	48.89 kB

The *Single SQL Statement Analysis* screen opens and shows the SQL statement as well as its prepared access plans:

```

1  SELECT /* Contains Native SQL */ "BOOKINGDATE", "BOOKINGDATE", "BOOKINGID", "BOOKINGID", "BOOKINGSTATUS", "BOOKINGSTATUS", "BOOKINGSTATUSTEXT"
2  "BOOKINGSTATUSTEXT", "CARRIERID", "CARRIERID", "CARRIERNAME", "CARRIERNAME", "CONNECTIONID", "CONNECTIONID", "CURRENCYCODE"
3  "CURRENCYCODE", "CUSTOMERNAME", "CUSTOMERNAME", "CUSTOMERNAME", "FLIGHTDATE", "FLIGHTDATE", "FLIGHTPRICE", "FLIGHTPRICE"
4  "LASTCHANGEDAT", "LASTCHANGEDAT", "TRAVELID", "TRAVELID"
5  FROM "DMOC_BKNG_PRCSR_M_51" "DMOC_BKNG_PRCSR_M_51"
6  WHERE "MAIND" = ?
7  AND ( "TRAVELID", "BOOKINGID" ) IN ( ?, ?, ? ) ) LIMIT ?
8  WITH PARAMETERS ( 'LOCALE' = 'EN' )
9
10
  
```

From the prepared plan, a HANA expert could now dig deeper and start analyzing the statement in detail.

## Results

With the drilldown options to all kinds of details that are all in one place in the technical monitoring cockpit, you're able to identify several possible reasons for the high HANA processing time and network time.

Are you interested in a "good" example of this kind of workload for comparison, with a server response time per request of under one second? The *ABAP Statistics Record* screen in such a case might look like this:

SAP Technical Monitoring Cockpit ▾ ABAP Statistics Record

ABAP Statistics Record  
13:46:42.35941200 +02:00 / 100 / 000000000730853637 / 75,472

Start Time: 13:46:42.35941200 +02:00 Program Context: /sap/opu/odata/DMO/UI\_TRAVEL\_PROC...  
End Time: 13:46:42.55709300 +02:00 Transaction Code:  
User Name: CB9980000481 Request Entry Name: /DMO/UI\_TRAVEL\_PROC\_M\_O\_S1/0001  
User Name (Extended Passport): CB9980000481 Request Entry Type: OData V2

Client: 100 Root Context ID: 161C47C3EC671EECABEABCDEBA2BCC68  
Tenant ID: 000000000730853637 Transaction ID: E4739B65E1E50050E006242ED6CDF325  
Record Number: 75,472 Session ID: 00002000161C47C3EC671EECABEABCDEBA...  
Communication: Encrypted Step Number: 2

Task Priority: H  
Task Type: HTTPS  
Work Process Type: dialog  
Work Process ID: 5

Server Response Time Contribution

100 -- 75,472 -- 2022-03-...

Legend: HANA Processing Time [ms] Database Network Time [ms] Load Times [ms] Enqueue Time [ms]  
Roll Times [ms] Roll Wait Time [ms] ABAP Processing Time [ms] Queue Wait Time [ms]

Time Component [ms]	Value
HANA Processing Time [ms]	27.69
Database Network Time [ms]	68.45
Load Times [ms]	15.24
Enqueue Time [ms]	2.51
Roll Times [ms]	0.29
Roll Wait Time [ms]	0
ABAP Processing Time [ms]	83.61
Queue Wait Time [ms]	0.03

SQL Statements

Statement	Database Processing Time	HANA Processing Time	HANA CPU Time	HANA Max. Memory	Executions	Records
CT /* Contains Native SQL */ "AGENCYID" "AGENCYID", "AGENCYNAME" "AGENCYNAME", "BEGINDATE" "B...	17.46 ms	14.71 ms	15.37 ms	137.55 kB	4	1 >
	9.22 ms	0.00 ms	0.00 ms	0.00 kB	54	0 >
CT "URL" FROM "V_UCONHTTPL_RT" WHERE "CLIENT" = ? AND ("HOST_NAME_LONG" = ? OR "HOST_NA...	6.20 ms	2.14 ms	1.84 ms	329.74 kB	3	509 >
CT TOP 1 HOST_CURRENT_CONNECTION FROM SYS.M_LANDSCAPE_HOST_CONFIGURATION WHERE IN...	5.94 ms	3.19 ms	1.97 ms	15.32 kB	4	2 >
RT INTO "RSAU_LOG" ("LOG_TSTMP", "EVENT", "SLGPROC", "SLGMAND", "SID", "INSTANCE", "COUNTE...	5.33 ms	2.02 ms	1.73 ms	14.09 kB	2	2 >
CT /* FDA READ */ "CLIENT" FROM "FTG_CUSTOM_T" WHERE "FEATURE_ID" = ?	4.79 ms	0.65 ms	0.30 ms	12.07 kB	5	0 >
CT "SERVICE_TIMESTAMP" FROM "SRVD_RT_EXTEENDS" WHERE "SERVICE" = ?	4.71 ms	0.69 ms	0.39 ms	28.09 kB	7	2 >
RT INTO "/DMO/LOG_TRAV_51" ("CLIENT", "CHANGE_ID", "TRAVEL_ID", "CHANGING_OPERATION", "CH...	4.14 ms	1.56 ms	1.36 ms	7.15 kB	2	2 >
RT INTO "/DMO/TRAQVL_M_51" ("CLIENT", "TRAVEL_ID", "AGENCY_ID", "CUSTOMER_ID", "BEGIN_DATE"...	3.89 ms	1.30 ms	1.05 ms	7.67 kB	2	1 >
RT INTO "/IWFND/ISU_STATS" ("MANDT", "ID", "TIMESTAMP", "SERVICE_NAME", "NAMESPACE", "VERSI...	3.58 ms	1.01 ms	0.80 ms	10.00 kB	2	2 >
	30.83 ms	5.37 ms	3.39 ms	65.71 kB	42	11 >

Database Connections

Connection Name	DB Requests ...	Database Processing Time	HANA Processing Time	HANA CPU Time	HANA Maximum Memory
DEFAULT	172	86.35 ms	27.69 ms	26.38 ms	329.74 kB >
	2	9.79 ms	0.00 ms	1.73 ms	14.09 kB >

Here, everything's fine: The server response time per request is only 0.197 seconds. The number of database requests is low, and therefore the database network time and processing times are low, too.

## Related Information

<https://blogs.sap.com/2023/04/24/analyzing-performance-degradations-in-the-abap-environment-in-the-cloud/>

### 6.3.3.1.3 Capturing ABAP Statistics Records to Analyze System Activities

Capture ABAP statistics records to find out which activities are running in your ABAP and database system.

#### ABAP Request Statistics

By default, in the ABAP environment, requests are recorded with technical information, such as response time, program name, or CPU time. The resulting individual ABAP statistics records are written to the file system of the relevant tenant. ABAP statistics records are well suited to investigating the performance or resource consumption of a system or of a tenant in more detail, or, if performance problems occur, to identifying the area causing the problems.

#### Use Cases

You typically use ABAP statistics records in the following scenarios:

- You want to investigate your system workload to find out about top consumers and investigate in detail the overall response time contributions or the database connections. In this case, you can use the ABAP statistics records that are captured by default and visualized in the [System Workload](#) app.  
For more information, see [Navigating from System Workload to Single ABAP Statistics Records](#).
- You want to analyze a specific workload, for example, the request statistics of an individual user or program name at a particular time.  
For more information, see [Capturing Request Statistics \[page 2937\]](#) and [Analyzing Captured Request Statistics \[page 2939\]](#).

## Related Information

[ABAP Statistics Records](#)

### 6.3.3.1.3.1 Capturing Request Statistics

With the *Capture Request Statistics* app, you can capture request statistics to find out which activities are running in your ABAP system.

#### Context

With the activation of a capture profile, you can capture system activities. Requests are recorded with technical information, such as response time, program name, or CPU time. Using the request statistics, you can investigate the performance or the resource consumption of a system or of a tenant in more detail.

In the *Capture Request Statistics* app, you can also find predefined profiles for capturing expensive requests (with names starting with SAP\_). These predefined capture profiles are active by default, and you can't edit or deactivate them.

You can view the captured data from all profiles in the technical monitoring cockpit on the *Captured Request Statistics* and *Request Processing* screens. You can get to the *Request Processing* screens by choosing *System Workload* from the menu. In addition, the captured ABAP statistics records are the basis for ABAP system sizing in the *Perform System Sizing* app.

#### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for the *Capture Request Statistics* app.  
The *Capture Request Statistics* app opens.
2. To create a new capture profile, choose *Create*.
3. Enter the following data:

Field	What to Enter
<i>Profile ID</i>	Enter a short, meaningful name for your profile. Don't use the prefix SAP_ because this prefix is reserved for profiles predefined by SAP.
<i>Description</i>	If needed, you can enter a longer description of your profile.
<i>Start/End Time</i>	The start and end time specify when request statistics were captured by the app. These fields are automatically filled after you have activated the profile.
<i>Retention Time</i>	Select how long you want to keep the captured request statistics stored in the system.
<i>Record Limit per Minute</i>	Enter the maximum number of records that you want the system to capture per minute.

Field	What to Enter
	<p>The record limit prevents that too much database memory is consumed by captured request statistics. The maximum possible record limit per minute is 1,000.</p> <p><b> ⓘ Note</b></p> <p>To get a realistic idea of the workload, consider a relatively high record limit. However, you must keep in mind that you generate load on your system while request statistics are captured.</p>
<i>Sampling Rate</i>	<p>Enter a sampling rate, which is the probability of capturing one single ABAP statistics record in percent.</p> <p>With the sampling rate, you ensure that a number of random records are selected during the capturing of request statistics.</p>
<i>Target User Group</i>	Choose one of the target groups starting with <i>Customer</i> .
<i>Profile Type</i>	This field is automatically populated by the system. Only static profile types (with fixed values) are possible.
<i>Profile Owner</i>	The profile owner is automatically set to <i>Customer</i> when you create a new profile. Predefined profiles have the profile owner <i>SAP</i> .
<i>Health Monitoring</i>	You can select the checkbox <i>Health Monitoring</i> for any capture profile that you create. As a result, the total number of captured ABAP statistics records for these profiles is shown in Health Monitoring of SAP Cloud ALM.

4. Choose *Create*.

The profile has been created, but is still inactive and has no filter conditions yet.

5. To define the filter conditions under which request statistics are captured, choose *Create*.
6. With *Field ID*, *Operator*, *High*, and *Low*, you can select a filter condition that determines what kind of data is captured. For example, you might be interested in requests with a high ABAP CPU or network time. Alternatively, you can also select a program name to capture requests relating to that program, for example

**ⓘ Note**

If you don't need to define a range (lower and upper limit), but only a single value, enter the value in the *Low* field.

For example, you can select *User ID* as field ID, *include equal to* as operator and your business user name to capture all request statistics relating to your business user.

7. Choose *Create*.

A new filter condition has been created.

8. To start capturing request statistics during the time period that's defined in the profile, choose *Activate*.

Make sure that you enter a start time that's right now or in the future. You can't activate profiles with a start time in the past.

## Results

Request statistics are captured and processed by a collector that runs every minute. Therefore, you must expect that captured request statistics are displayed with a delay of a minute in the [System Workload](#) or in the [Perform System Sizing](#) app. Capturing request statistics is finished when the status of relevant capture profile has changed to [Finished](#).

### ⓘ Note

If you have a long-running request, ABAP statistics records for this request will only show up in the technical monitoring cockpit after the request is done. The collector then captures the request and calculates its workload over its runtime.

### 6.3.3.1.3.2 Analyzing Captured Request Statistics

Get an overview of the request statistics that you have captured using the [Capture Request Statistics](#) app. View the server response time contribution or the database connections of a single request.

## Context

With the [Capture Request Statistics](#) app, you can define conditions for which request statistics are collected in the system (see [Capturing Request Statistics \[page 2937\]](#)). You can use the technical monitoring cockpit to view the results.

### ⓘ Note

Before ABAP statistics records are shown in the technical monitoring cockpit, they are captured and processed by a collector that runs every minute, so you must expect that captured request statistics are displayed with a delay.

## Procedure

1. In the [Capture Request Statistics](#) app, choose a profile for which you want to analyze the captured data.
2. Choose [Analyze Captured Request Statistics](#).

The [Captured Request Statistics](#) screen opens in the technical monitoring cockpit. The captured request statistics shown are filtered; only the results for the selected profile are shown.

- Analyze the data for different aspects like identifying expensive requests and their context (program name, transaction code, or path of an HTTP request), so that you can find the area causing performance issues or resource shortages.

For more information, see [Captured Request Statistics](#).

### 6.3.3.1.4 Monitoring Expensive Outbound Communication

The [Capture Request Statistics](#) app and the technical monitoring cockpit, optionally in combination with SAP Cloud ALM, help you to identify service requests that cause expensive outbound communication (RFC, HTTP, and Web service calls).

#### Use Cases

You might want to find out what the requests with the most expensive system outbound communication are:

- Which communication arrangements have a long calling time?
- How is the calling time distributed among communication scenarios and how does it behave over time?
- Are there any outliers?

After you have found answers to these questions, you might want to dig deeper into the details of one particular communication arrangement.

Alternatively, you might already have a good idea which communication arrangements are associated with a longer calling time. For example, you notice that one of your customer applications does not run smoothly because the related system communication takes a long time.

In both cases, you want to find out the root cause for the long calling times. Therefore, you might want a heads-up when the long calling times occur. In addition, you also need to find the affected service requests and you want to collect their ABAP statistics records.

In these cases, a combined use of the technical monitoring cockpit, the [Capture Request Statistics](#) app and SAP Cloud ALM is beneficial.

#### Process Flow

- If you want to get an overview of the system outbound communication first, use the technical monitoring cockpit to find expensive service requests and their related communication arrangements (see also [Monitoring System Outbound Communication in General \[page 2941\]](#)).  
In the technical monitoring cockpit, the top 10 communication scenarios by calling time are shown, which helps you identify obvious expensive outbound communication. If you have identified a particular communication arrangement, you can now get more detailed data.  
However, sometimes the issues during outbound communication occur too infrequently, so you might miss them with only an occasional check. It might also happen that critical service requests do not appear in the top 10 communication scenarios by calling time, or they do, but the related detailed statistics are not part

of the samples that are collected by default for the technical monitoring cockpit. Therefore, you now turn to the [Capture Request Statistics](#) app to collect more specific, user-defined data.

2. To collect ABAP statistics records relating to expensive outbound communication, for example, for a specific communication arrangement or for particularly long calling times, define a profile in the [Capture Request Statistics](#) app. Make sure that the [Health Monitoring](#) checkbox is selected in your profile (see [Capturing Request Statistics Relating to Expensive Outbound Communication \[page 2944\]](#)).
3. In Health Monitoring in SAP Cloud ALM, use the metric [Captured Request Statistics](#) to find out whether service requests exceed a defined threshold in their calling time (see [Getting Alerted About Expensive Outbound Communication Using SAP Cloud ALM \[page 2946\]](#)). You can also use the settings in SAP Cloud ALM to get alerted with an e-mail notification.
4. After you have received a notification, use the [Captured Request Statistics](#) screen in the technical monitoring cockpit to analyze the captured request statistics for outbound communication (see [Analyzing Expensive Outbound Communication Using the Technical Monitoring Cockpit \[page 2948\]](#)).

### 6.3.3.1.4.1 Monitoring System Outbound Communication in General

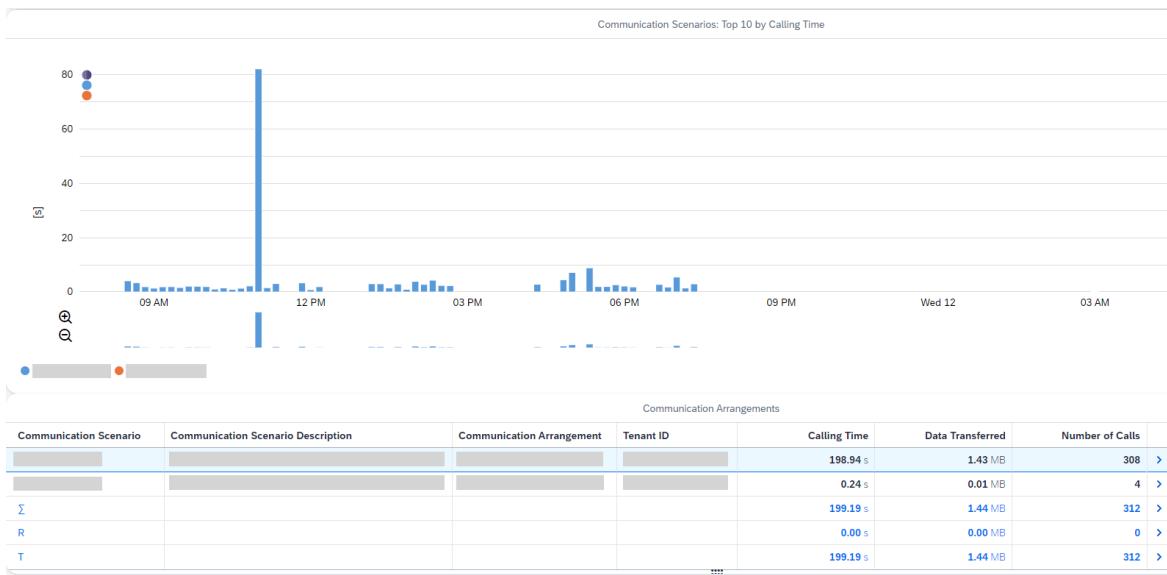
Get an overview of the system outbound communication over time.

#### Context

In the technical monitoring cockpit, the top 10 communication scenarios by calling time are shown, which helps you identify obvious expensive outbound communication. The focus is on outbound communication of type RFC, HTTP, and Web service.

#### Procedure

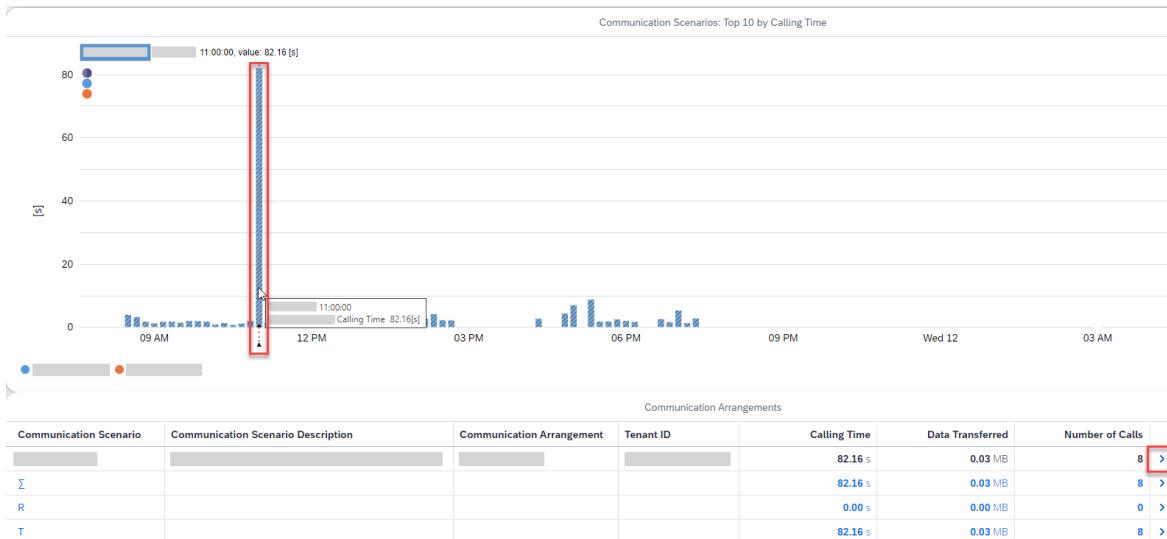
1. On the SAP Fiori launchpad of your ABAP environment, search for [Application System Overview](#).  
The technical monitoring cockpit opens.
2. On the dashboard shown, choose the [System Outbound Communication](#) tile.  
The top 10 communication scenarios by calling time are shown, including the related communication arrangements. In the chart, the development over time is shown. Only customer workload using customer communication arrangements is displayed; SAP workload and SAP communication arrangements are filtered out.



- In the chart, you can also click on a communication scenario at a particular time.

As a result, the table of communication arrangements is updated with the single communication arrangements at this particular time.

In this rather quiet test system, workload is low, but one communication scenario before 12 p.m. in the chart stands out with a long calling time.

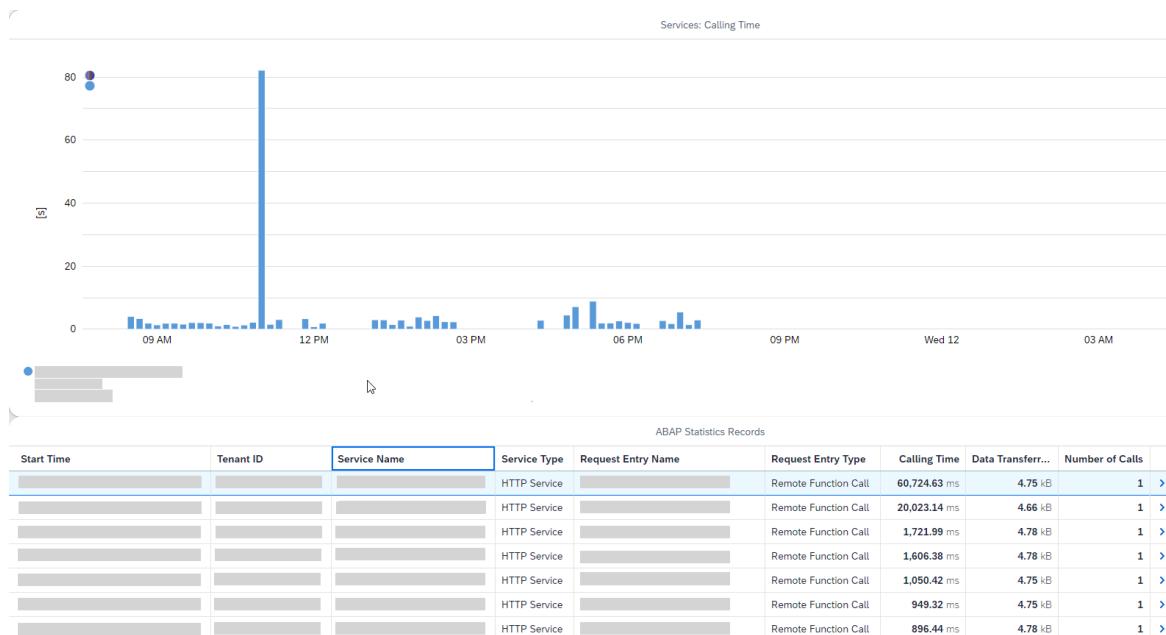


- To analyze such a specific communication arrangement in more detail, choose its entry from the list of communication arrangements.

The *Communication Arrangement* screen opens, where the selected communication arrangement with its related outbound services is shown. The chart indicates how the outbound services perform over time. You can now check whether calling time is distributed evenly or whether there are any outliers.

In the list, you can find the single ABAP statistics records for the top outbound-calling services. The records include the calling time, the amount of transferred data and the number of calls.

Again, in the example shown here, there's an outlier among the services that merits further investigation.



- To show more details of a single ABAP statistics record, choose it from the list.

As a result, you navigate to the *ABAP Statistics Record* screen, where more details, such as the server response time and multiple subrecords are shown.

- To see the details of the HTTP subrecord, choose *HTTP*.

Here, you can find the single outbound HTTP client calls that were made during the service request.

- To drill down to even more details, choose an entry from the table of HTTP client calls.

You now get the HTTP client call details, including its calling time, and also the related communication arrangement.

## Results

The *System Outbound Communication* screen and its related drill-down screens are configured by default in such a way that they provide sample records of expensive system outbound communication. If you want to get a full set of ABAP statistics records for a particular communication arrangement with a long calling time, use the *Capture Request Statistics* app. With this app, you can capture request statistics relating to defined expensive outbound communication arrangements.

## Related Information

[Capturing Request Statistics Relating to Expensive Outbound Communication \[page 2944\]](#)

[System Outbound Communication](#)

## 6.3.3.1.4.2 Capturing Request Statistics Relating to Expensive Outbound Communication

Use the [Capture Request Statistics](#) app to collect statistics of service requests with an expensive outbound communication (RFC, HTTP, or Web service).

### Prerequisites

You need a system administrator role (based on SAP\_BR\_ADMINISTRATOR) to perform these tasks.

### Context

With the [Capture Request Statistics](#) app, you can capture single ABAP statistics records of dedicated workloads in your ABAP system (see also [Capturing Request Statistics \[page 2937\]](#)). You can also use this app for monitoring expensive outbound communication. For this purpose, you configure a capture profile to monitor the calling time of outbound calls of type HTTP, RFC, and Web service that exceed a defined threshold. In this procedure, you can capture records related to a high calling time. Alternatively, you might already have an idea which communication arrangements are related to expensive outbound communication, and you would like to find out the corresponding service requests.

With the checkbox [Health Monitoring](#) selected, you can also automatically export the number of captured ABAP statistics records of the profile to Health Monitoring in SAP Cloud ALM so that you get alerted when expensive outbound communication occurs.

### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for the [Capture Request Statistics](#) app.
2. Choose [Create](#) and enter the basic header information for the profile.

If you want to export the number of capture request statistics to SAP Cloud ALM, select the [Health Monitoring](#) checkbox.

For more information about how to fill out the individual fields in the header, see the built-in SAP Companion documentation.

3. To define a filter for high outbound calling times, choose [Create](#) and enter the following data:

Option	Description
Field ID	Choose <a href="#">Calling Time [ms]</a> from the dropdown list.
Operator	Select <a href="#">Include greater than or equal to</a> .

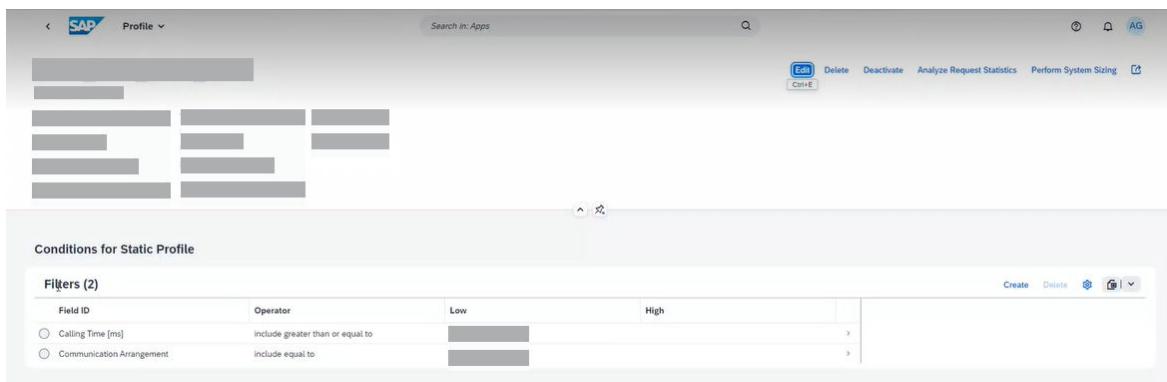
Option	Description
	<p><b>① Note</b></p> <p>When you enter a metric in the <i>Low</i> field, it doesn't matter which operator you choose here because it's always interpreted as threshold. The operator is therefore always interpreted as <i>Include greater than or equal to</i>.</p>
<b>Low</b>	Enter a threshold for the calling time, for example, 1000. As a result, data for all service requests is captured whose calling time exceed 1,000 ms.
<b>High</b>	Leave this field empty.

- Choose [Create](#) to save the filter and go back to the header data.

If you don't want to filter for any specific communication arrangement, you can stop defining filters here and activate your capture profile. The app will then capture any service request that makes at least one outbound RFC, HTTP, or Web consumer call and whose **total** calling time for all calls of each type exceeds the defined threshold.

- To define a filter for a specific communication arrangement, choose [Create](#) and enter the following data:

Option	Description
<b>Field ID</b>	Select <i>Communication Arrangement</i> from the dropdown list.
<b>Operator</b>	Select <i>Include equal to</i> .
<b>Low</b>	From the dropdown list, select the communication arrangement that you want to monitor.
<b>High</b>	Leave this field empty.



- Choose [Create](#) to save the filter and go back to the header data.

Optionally, you can define more filters for additional communication arrangements.

If you have also defined filters for communication arrangements, the [Capture Request Statistics](#) app will then only consider the service requests for the communication arrangements in the filters. In addition, data for a request using one of these communication arrangements is only captured if the total of at least

one of the request's outbound RFC, HTTP, or Web consumer calls exceeds the defined threshold for the calling time.

7. Activate the capture profile.

### 6.3.3.1.4.3 Getting Alerted About Expensive Outbound Communication Using SAP Cloud ALM

If you use SAP Cloud ALM to monitor your applications in a Cloud system landscape, you can also use health metrics to get alerted about expensive outbound communication.

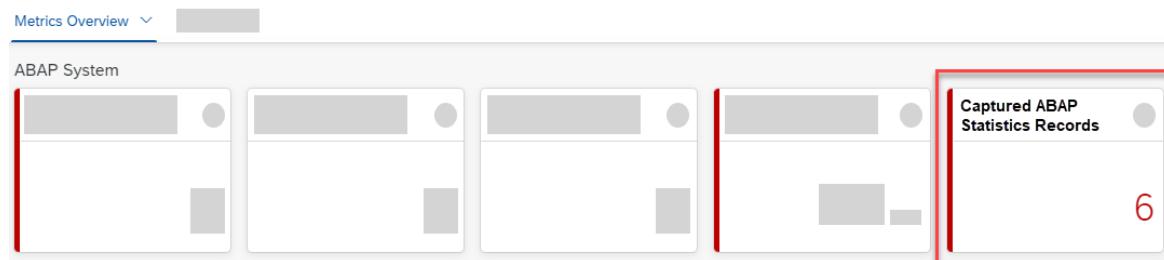
#### Prerequisites

You have SAP Cloud ALM set up (see [Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#)).

You have captured request statistics relating to outbound communication using the *Capture Request Statistics* app, and you have selected the checkbox *Health Monitoring* in the relevant capture profiles (see [Capturing Request Statistics Relating to Expensive Outbound Communication \[page 2944\]](#)).

#### Procedure

1. Log on to SAP Cloud ALM and call up Health Monitoring.
2. On the Health Monitoring dashboard for your ABAP system, check the metric [Captured ABAP Statistics Records](#).



The number of captured ABAP statistics records that is shown in SAP Cloud ALM is the total number of captured records for profiles with the checkbox *Health Monitoring* selected.

3. If the number of records is above zero, you have a clear indication that there was expensive outbound communication in the ABAP system during the last 5 minutes.
4. Choose the [Captured ABAP Statistics Records](#) tile and then *History*.

On the *History* popup, you can find out how many records were captured by the profile that you've chosen and how the number of records changed over time.

## ⓘ Note

A high number of captured ABAP statistics records with a high calling time are an indication that the performance of outbound communication gets critical. Use the settings in SAP Cloud ALM if you want to get notified with an e-mail as soon as a critical number of ABAP statistics records have been captured.

**SAP Cloud ALM Notification**  
Health Monitoring Event

**Critical Workload Records Captured**

Service Name	[REDACTED]
Service Type	SAP BTP ABAP Environment
Object Details	[REDACTED]
Reported at	11-Oct-[REDACTED] 11:13:44
Rating	Critical

**Additional Information**

Name	Profile	Timestamp	Rating	Value
Number of Captured Requests	[REDACTED]	11-Oct-[REDACTED] 11:10:00	Critical	6.0

[Click here to view this alert](#)  
[Click here to view the service/system in Health Monitoring](#)

This is an automated email, please do not reply.  
If you do not want to receive further notifications from SAP Cloud ALM, you can click to [unsubscribe](#).

Let's Run Better Together!  
**Your SAP Cloud ALM Team**



## Results

You can now go to the technical monitoring cockpit to analyze the expensive outbound communication in more detail.

## Related Information

[Analyzing Expensive Outbound Communication Using the Technical Monitoring Cockpit \[page 2948\]](#)

## 6.3.3.1.4.4 Analyzing Expensive Outbound Communication Using the Technical Monitoring Cockpit

With the technical monitoring cockpit, you can analyze expensive outbound communication in detail.

### Prerequisites

You have set up a profile to capture request statistics relating to outbound communication using the *Capture Request Statistics* app (see [Capturing Request Statistics Relating to Expensive Outbound Communication \[page 2944\]](#)).

### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for the *Capture Request Statistics* app.
2. From the list of capture profiles, choose the profile that you have defined to collect ABAP statistics records for expensive system outbound communication.
3. Choose the *Analyze Request Statistics* button.

The *Captured Request Statistics* screen in the technical monitoring cockpit opens. On this screen, you can find all workloads that were captured using your profile in the *Capture Request Statistics* app. You can now investigate in more detail which service requests were responsible for expensive outbound communication and what details their ABAP statistics records contain, such as the related outbound HTTP, RFC, or Web service calls.

### Related Information

[ABAP Statistics Records](#)

## 6.3.3.1.5 Analyzing the Performance of SQL Statements Running on Your ABAP Environment

Learn how you can use SQL statement and trace analysis in the ABAP environment.

Data retrieval and modification via SQL statements is the main business of the SAP HANA database in your ABAP Environment. When you develop new applications, or when you operate applications productively, you want to analyze the performance of the SQL statements executed by your applications on the database.

When you analyze statement performance, you typically have one of the following scenarios:

- You are only interested in the performance of SQL statements of your application regardless of other SQL activity on your database.

For this, you can use the SQL trace analysis in the technical monitoring cockpit to display and analyze SQL traces you have run with your ABAP Development Tools.

- You want to see the performance of SQL statements alongside the rest of the SQL activity on your database. The right starting point for this is the SQL statement analysis of the technical monitoring cockpit.

For each of the scenarios, we show you how to proceed here.

### 6.3.3.1.5.1 Analyzing SQL Statements of Your Application

#### Context

##### ⓘ Note

The following procedure is for developers. If you have the role of an administrator, see [Tracing SQL Activities for Administrators \[page 2953\]](#).

During developing and operating applications on your ABAP environment, these situations are typical when you take an SQL trace of your ABAP application and turn to the SQL trace analysis:

- You want to analyze the performance of the SQL statements of your ABAP application in the sequence as they run.
- When analyzing the performance of an ABAP application, you want to clarify whether the SQL activity of the application is a relevant contribution to the processing time.

If you're familiar with on-premise products from SAP and know the SAP GUI transaction ST05 (performance trace), you'll be happy to see that the SQL trace functions described in the following procedure are similar to what you know from the SQL trace in transaction ST05.

#### Procedure

1. Take an SQL trace of your ABAP application using ABAP development tools (ADT):
  - a. Right-click on your ABAP project on the project explorer and choose [SQL Trace](#) from the context menu.
  - b. Activate the trace.
  - c. Run the ABAP application that you want to trace.
  - d. Deactivate the trace again and choose [View Trace Directory](#).

This opens the [SQL Trace Analysis](#) screen of the technical monitoring cockpit.

SQL Trace Analysis						
Trace Directory	Trace Records	SQL Statement	Prepared Plan	Executed Plan		
>  12:08  +02:00 (Local Time)	<  1D As Collected >			12:08		
>	Characteristics (7/10) Metrics (0/0) Filter (1) Sort (1) Rows (≤1000) <b>Go</b> ...					
Rows (3)	Show Chart	Pivot	...			
Time Stamp	Client	Owner	Instance Name	Expiry Date	Description	
14:04:46 +02:00	000		_33	0000-00-00		>
14:04:46 +02:00	000		_34	0000-00-00		>
14:04:46 +02:00	000		_33	0000-00-00		>

2. On the *Trace Directory* tab page of the SQL trace analysis, choose the new trace you just took.

The default view shows you all SQL traces taken by your ABAP user in the last 24 hours.

The UI philosophy of the *SQL Trace Analysis* screen is: With each move to a tab page to the right, you get more information about what you've selected on the previous tab pages to the left.

3. Choose the trace that you want to look at and change to the *Trace Records* tab page.
4. On the *Trace Records* tab page, single out expensive or otherwise interesting SQL statements.

By default, you see all trace records in chronological order. However, you can apply filter conditions, for example, on the SQL statement text, or sort the entries by duration to put the longer running statements to the top of the list. Note that you can display additional columns that aren't shown by default by choosing *Characteristics* or *Metrics*.

To go further into detail from here, you have three options on the following three tab pages to the right (*SQL Statement*, *Prepared Plan*, *Executed Plan*), which are mostly independent of each other. You can, for example, jump from here directly to the calculation of an executed access plan for your statement.

5. Choose the *SQL Statement* tab page to get a decent view of your selected SQL statement, pretty-printed and with syntax highlighting.
- The *Application Source Position* field shows you the ABAP code location (if available) where the SQL statement originates from. You can choose one of the *Show ABAP Source in ...* buttons to display the ABAP source in the web browser or in ABAP Development Tools.
6. Choose the *Prepared Plan* tab page to display and analyze a newly calculated prepared access plan for your SQL statement.
  7. Choose the *Executed Plan* tab page to display and analyze a newly calculated executed access plan for your SQL statement.

You can also download the PlanViz (PLV) file of the executed access plan containing the comprehensive plan information to display it with other HANA plan visualizing tools like HANA SQL Analyzer.

## 6.3.3.1.5.2 Analyzing SQL Statements in the Context of What Else Is Going On on the Database

### Context

When developing and operating applications on your ABAP environment, you turn to the SQL statement analysis typically in the following situations:

- You want to see the performance of one SQL statement of your application alongside the rest of the SQL activity on your database for a given time period.
- You're planning to identify the expensive SQL statements in your system to find candidates that merit the effort of performance tuning.

In the second situation, your starting point isn't a particular SQL statement but you have a look at the whole set of statements running on the database. The most important metric to look at is the share of total execution time. However, you might also like to look for a large lock-wait time, if locking is an issue (for example, for data changing statements). Choosing specific time ranges can help to highlight SQL statements that are expensive when they run but don't run often. They can also show a large average execution time while the total execution time on a larger time scale isn't that prominent in the top 25.

Once you've identified expensive SQL statements, which you would like to analyze in more detail to find potential for performance improvement. In any case, you also end up with analyzing single statements in the end.

### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for [SQL Statement Analysis](#).

The technical monitoring cockpit opens with the [SQL Statement Analysis](#) screen.

2. Use the sorting and filtering functions of the technical monitoring cockpit to get what you want:

If the statement you want to analyze isn't among the 25 most expensive statements in the last 24 hours, use sorting, filtering, or the time slider to find it. For example, you can change the time period, set filters on the statement string, or sort by another metric to highlight other aspects of SQL statement execution. On the [Plan Cache](#) tab page, you can also display more SAP HANA plan cache characteristics and metrics than shown by default by choosing [Characteristics](#) or [Metrics](#) and following the selection dialog.

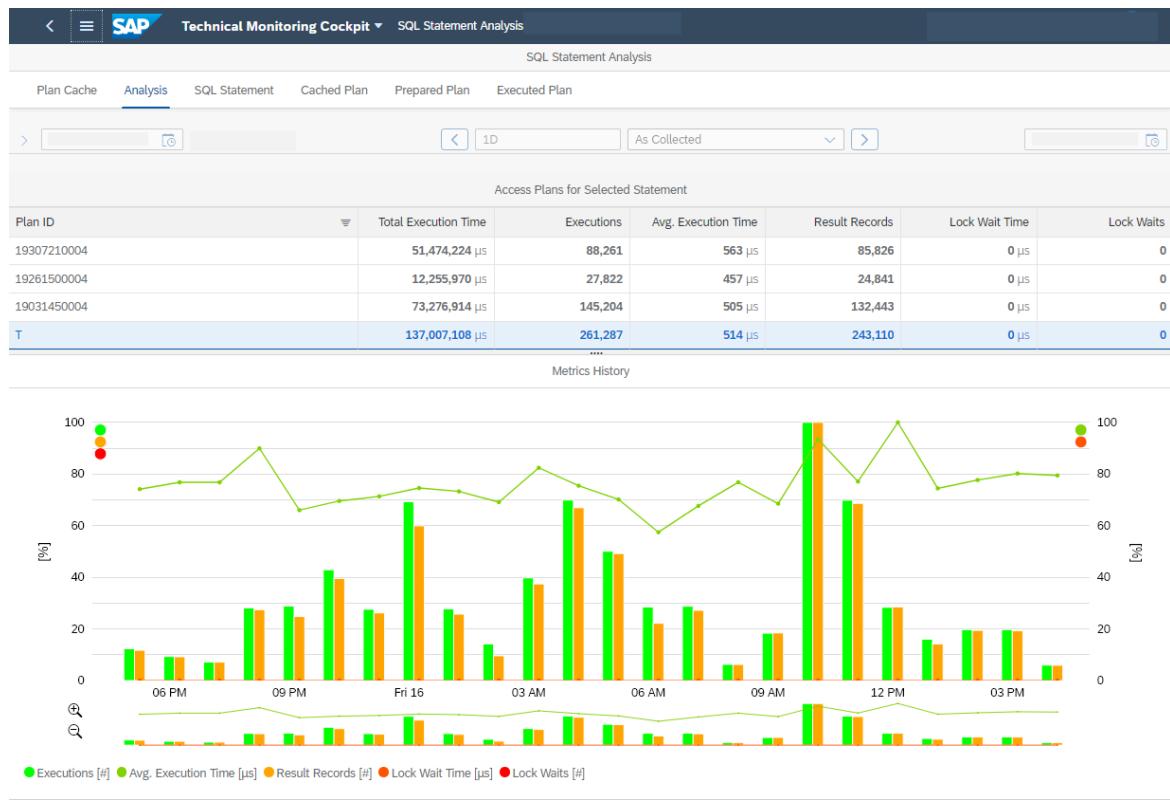
Once you've found your statement in the list, you can put the performance of your statement into perspective. Compare it to other statements appearing in the list, but also against the accumulated performance of all the rest of the statements matching your filter criteria. The accumulated performance of all statements not matching your filter criteria is subsumed in the bottom row, labeled with the statement string [R](#).

3. Choose the statement that you are interested in.

As a result, all subsequent tabs to the right of [Plan Cache](#) now contain detailed information on the statement that you have selected.

- Choose the [Analysis](#) tab page to see how your statement's execution performance has developed over time.

On the screen area [Access Plans for Selected Statement](#), you can see whether your statement has entered the HANA plan cache with different plan IDs in the selected time range. The bottom row, labeled with plan ID *T*, shows you the metrics totals, that is, the aggregated values.



Seeing more than one plan ID often only means that the access plan of a statement was evicted from the plan cache at some point and reentered it later. However, it can sometimes also indicate a change of the access plan of your statement with implications for its performance. As you get the same performance metrics as on the [Plan Cache](#) tab page separately for each plan ID, you can check whether the average execution time has significantly changed with plan ID. If this is the case, you may be on to something.

On the [Metrics History](#) screen area, you can see the time dependence of the major performance metrics in relative units. You get the absolute value of a metric by hovering over a datapoint.

When you choose a particular plan ID in the upper screen area, you'll see the performance metrics of the execution of your chosen SQL statement with this plan ID only. When you choose the totals row (plan ID *T*), the overall performance metrics of the statement are shown, regardless of the plan ID.

If you want to go further into detail from here, you can use the tabs [SQL Statement](#), [Cached Plan](#), [Prepared Plan](#), and [Executed Plan](#), which are mostly independent of each other. You can, for example, jump from here directly to the calculation of an executed access plan for your statement. However, we follow here the order of the tabs in the program.

- Choose the [SQL Statement](#) tab page to get a decent view of your selected SQL statement, pretty-printed and with syntax highlighting.

The [Application Sources](#) dropdown list shows you the list of ABAP code locations (if available) where the SQL statement originated from in the selected time range. Choose an ABAP location from the list and

choose one of the *Show ABAP Source in ...* buttons to display the ABAP source in the web browser or ABAP Development Tools.

You can also edit the SQL statement and recalculate the prepared or executed access plan. In this way, you can try out if changes to the statement or adding optimizer hints have a positive or negative impact on the access plan. You can iterate this procedure. Note though that the information on the tab page *Analysis* corresponds to the SQL statement that you selected originally on the *Plan Cache* tab page. There is no cached access plan for your edited statement on the *Cached Plan* tab page, obviously. Once you've edited the SQL statement, you can get back to the original statement by using the undo function (CTRL+Z) of your browser while having focused on the statement display area. Alternatively, you can go back to the *Plan Cache* tab page and choose the relevant SQL statement there.

Note that the SQL statement analysis doesn't support SQL query parameters.

6. Choose the *Cached Plan* tab page to display and analyze the prepared plan that has been calculated when your SQL statement entered the HANA plan cache with the selected Plan ID from tab page *Analysis*.
7. Choose the *Prepared Plan* tab page to display and analyze a newly calculated prepared access plan for your SQL statement.
8. Choose the *Executed Plan* tab page to display and analyze a newly calculated executed access plan for your SQL statement.

You can also download the PlanViz (PLV) file of the executed access plan containing the comprehensive plan information to display it with other SAP HANA plan visualizing tools like HANA SQL Analyzer.

### 6.3.3.1.6 Tracing SQL Activities for Administrators

With the *Capture Request Statistics* app, you can activate and capture SQL traces for a later analysis.

#### Context

The *Capture Request Statistics* app allows you to trace the SQL activity of your ABAP environment from the SAP Fiori Launchpad if you're in the role of an administrator (business role template SAP\_BR\_ADMINISTRATOR).

To run an SQL trace, you create a profile of type *SQL Trace* and activate it (the app does not offer predefined SQL trace profiles). All SQL activities of your ABAP environment that fulfill the filter conditions defined in the profile are then recorded until you deactivate the profile. You can view the trace results on the *SQL Trace Analysis* screen in the technical monitoring cockpit.

#### Procedure

1. On the SAP Fiori launchpad of your ABAP environment, search for the *Capture Request Statistics* app and open it.

On the first page of the app, you'll find a list of existing profiles. The relevant profiles for your use case are capture profiles of type *SQL Trace*.

2. To create a new capture profile, choose *Create*.
3. Enter the following data in the header section that opens:

Field	What to Enter
<i>Profile ID</i>	Enter a short, meaningful name for your profile. Don't use the prefix SAP_ because this prefix is reserved for profiles predefined by SAP.
<i>Description</i>	If needed, you can enter a longer description of your profile.
<i>Target User Group</i>	Choose one of the target groups starting with <i>Customer</i> .
<b>Profile Type</b>	Choose <i>SQL Trace</i> .

4. To confirm your input, choose *Create*.

Now the profile is created, but is still inactive and has no filter conditions yet apart from a filter on your own user name, which you can keep or remove according to your preferences.

5. To define filter conditions under which SQL statements are captured by the profile, choose *Create* in the *Conditions for SQL Trace* section.
6. In the dialogue that opens, choose a *Field ID* from the dropdown list, such as *CDS View or Table*, *Program Name*, or *User Name*. Also choose an *Operator* (for example, *include equal to*), and in the *Low* field, enter your specific filter value.

#### ① Note

You can set almost all conditions only with the *include equal to* operator. Only for the field *CDS View or Table* can you choose between *include* and *exclude equal to*.

There's a limit for the number of operators:

For *CDS View or Table*, you can enter up to **five** *include equal to* and up to five *exclude equal to* conditions. Such a combination results in a hit if one of the *include* and one of the *exclude equal to* conditions match. It's possible to use a wildcard (\*).

For *Program Name*, *RFC Function Name*, and *Transaction Name*, you cannot set multiple conditions, but only one per profile.

7. Choose *Create*.

A new filter condition has been created. If you want to set more filter conditions for the profile, repeat steps 5 to 7.

8. To finally start the SQL trace, choose *Activate*.

The following popup tells you whether another user is currently running an SQL trace in the system. If so, you cannot start a new one because it's only possible to run one trace at a time. The running trace must first be stopped before you can start a new one.

### ⓘ Note

You cannot stop a running trace if it was started on a different tenant. You can only stop running traces of users of your own user group.

9. To stop SQL tracing, choose *Deactivate*.
10. To view and analyze the captured SQL statements for your profile, choose *View Trace Directory*.

This leads you to the *SQL Trace Analysis* in the technical monitoring cockpit. If you activate an SQL trace profile that you've previously run, you'll trigger a new trace run, which is then available again on the *SQL Trace Analysis* screen as a new set of trace records.

## 6.3.3.2 Monitoring the System Health

With health monitoring metrics, you can check the health of your ABAP system from an application and customer perspective.

Metrics are collected on a regular basis and give you an idea of the system health with a special focus on performance, events, configuration, and the quota used by the ABAP system.

For the ABAP environment, you can choose between central monitoring tools such as SAP Cloud ALM or SAP Focused Run and an SAP Fiori app for local health monitoring:

	SAP Cloud ALM or SAP Focused Run	SAP Fiori App
Use cases	Monitoring multiple systems in a customer system landscape	Monitoring the local ABAP system
Prerequisites and constraints	SAP Cloud ALM or SAP Focused Run available	Local monitoring of only the system where the user is logged on
Benefits	<ul style="list-style-type: none"><li>• Display of health metrics for multiple systems</li><li>• Alerts and notifications</li></ul>	Overview of metrics on one single screen

In addition to metrics provided by SAP, you can also define your own metrics for health monitoring using ABAP Development Tools. You can make these metrics automatically available in SAP Cloud ALM or SAP Focused Run.

## Related Information

[Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#)

[Local Health Monitoring \[page 2963\]](#)

[Health Monitoring Using Your Own Metrics \[page 2968\]](#)

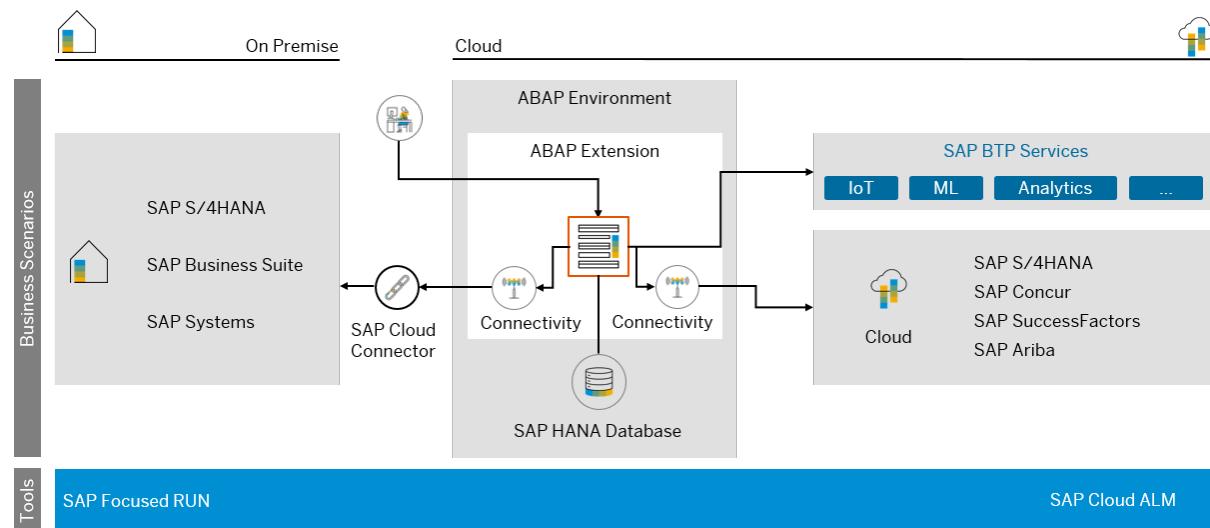
## 6.3.3.2.1 Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM

Learn more about how you can monitor the ABAP environment using SAP Focused Run and SAP Cloud ALM.

### Context

If you run SAP BTP, ABAP environment, as part of a hybrid landscape with on-premise and cloud systems, you might already have a central monitoring and alerting infrastructure in place. For example, the service offering SAP Focused Run is designed specifically for businesses that need high-volume system and application monitoring, alerting, and analytics, especially for hybrid landscapes including on-premise SAP solutions.

In addition, for cloud-centric customers, there's SAP Cloud ALM as an application lifecycle management offering.



If you use SAP Focused Run or SAP Cloud ALM as monitoring and alerting infrastructure, you can integrate monitoring of the ABAP environment into your existing infrastructure. Using the health monitoring in SAP Focused Run or SAP Cloud ALM, you can watch whether your ABAP environment is still up and running and whether any exceptional situations occurred. In addition, you can use the real-user monitoring to monitor requests coming from business users and integration monitoring to watch the communication between integrated systems.

## Available Metrics for the ABAP Environment

In the health monitoring of SAP Focused Run and SAP Cloud ALM, the following metrics for application checks are available for the ABAP environment:

Metrics for the ABAP Environment in Health Monitoring (Application Check)

Area	Metric Label	Technical Metric Name	Description
ABAP System	ABAP Runtime Errors	abap_core_dump_count_5m	The number of ABAP runtime errors during the last 5 minutes
ABAP System	Current Unique Users	abap_system_users_count	The number of current unique users in the ABAP system
ABAP System	Locked Business Users	abap_system_locked_bus_usr_count	The number of currently locked business users
ABAP System	Locked Communication Users	abap_system_locked_com_usr_count	The number of currently locked communication users
ABAP System	Current Sessions	abap_system_sessions_count	The number of current sessions in the ABAP system
ABAP System	Captured ABAP Statistics Records	abap_system_ksr_captured_count_5m	The number of ABAP statistics records that were captured by the <a href="#">Capture Request Statistics</a> app during the last 5 minutes
<p>In the <a href="#">Capture Request Statistics</a> app, you can select the checkbox <a href="#">Health Monitoring</a> for any capture profile that you create. The number of records that is shown in SAP Cloud ALM is the total number of captured ABAP statistics records during the last 5 minutes for profiles with the checkbox <a href="#">Health Monitoring</a> selected.</p> <p>For more information about one use case of this metric, see <a href="#">Monitoring Expensive Outbound Communication</a> [page 2940].</p>			

Area	Metric Label	Technical Metric Name	Description
ABAP System	ABAP Compute Units	abap_acu_used_count_5m (type: memory)	<p>Used quota for ABAP system memory during the last 5 minutes</p> <p>A quota represents the available system size and therefore the maximum allowed consumption of a resource. A resource is measured against the quota independently. For ABAP system resources, the quota is measured in ABAP compute units (ACUs).</p>
ABAP System	ABAP Compute Units	abap_acu_used_count_5m (type: cpu)	<p>Used quota for ABAP CPU utilization during the last 5 minutes</p> <p>A quota represents the available system size and therefore the maximum allowed consumption of a resource. A resource is measured against the quota independently. For ABAP system resources, the quota is measured in ABAP compute units (ACUs).</p>
ABAP System	ABAP Compute Units	abap_acu_used_count_5m (type: allocation)	<p>Resources (CPU and memory) that were allocated to your ABAP system during the last five minutes</p> <p>For ABAP system resources, the allocation is measured in ABAP compute units (ACUs). In a static service plan, the allocation corresponds to the quota limit of the service plan. In an elastic service plan, the allocation of resources is handled in an adaptive way.</p>

Area	Metric Label	Technical Metric Name	Description
ABAP System	Expiry of Client Certificates	abap_system_client_cert_expiry_d	<p>Expiry of client certificates in days</p> <p>With this metric, you can monitor whether any client certificates expire that you have uploaded to the ABAP system for the communication with systems or services outside the ABAP system, for example, BTP services. Make sure that your client certificates are valid, so that communication doesn't break off.</p> <p>If the client certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>
ABAP System	Expiry of Communication System Certificates	abap_system_comsys_cert_expiry_d	<p>Expiry of communication system certificates in days</p> <p>With this metric, you can monitor whether any communication system certificates uploaded to the ABAP system expire for the communication with systems or services outside the ABAP system. Make sure that your communication system certificates are valid, so that communication doesn't break off.</p> <p>If the communication system certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>

Area	Metric Label	Technical Metric Name	Description
ABAP System	Expiry of Trust List Certificates	abap_system_trusts_cert_expiry_d	<p>Expiry of trust list certificates in days</p> <p>With this metric, you can monitor certificates from communication partners that you have added to the certificate trust list in your ABAP system. The metric shows whether any of these trusted certificates expire for the communication with systems or services outside the ABAP system. Make sure that your trusted certificates are valid, so that communication doesn't break off.</p> <p>If the trusted certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>
ABAP System	Critical Number Range Intervals	abap_system_nr_critical_interval_pct	<p>Fill ratio of number range intervals</p> <p>Only critical number range intervals are shown. When you create a number range interval, you can define the critical remaining fill ratio when a warning is generated.</p> <p>If a critical number range was reported and is extended, it's reported once again with its now uncritical, good fill ratio to report the fixed state. After that, this number range is dropped from reported number range intervals because it's not critical anymore.</p>
ABAP System	E-Mail Transmissions (Last Day)	abap_system_email_error_count_1d	The total number of e-mails and the number of e-mails with errors during the last day

Area	Metric Label	Technical Metric Name	Description
ABAP System	E-Mail Transmissions (Last 5 Minutes)	abap_system_email_error_count_5m	The total number of e-mails and the number of e-mails with errors during the last 5 minutes
ABAP System	Table Change Deltas	abap_system_bc_dbatalog_delta_count_1d	Table changes during the last day per database table for database tables with more than 10,000 changes
ABAP System	Table Changes	abap_system_bc_dbatalog_count_1d	Total table changes during the last day per database table for database tables with more than 50,000 changes
HANA	HANA Out-of-Memory Events	hana_db_oom_event_count_5m	The number of out-of-memory events on the SAP HANA index server during the last 5 minutes
HANA	HANA Compute Units (CPU)	hana_hcu_used_count_5m (type: cpu)	Used quota for CPU utilization during the last 5 minutes  A quota represents the available system size and therefore the maximum allowed consumption of a resource. Each resource is measured against the quota independently. For SAP HANA database resources, the quota is measured in HANA compute units (HCUs).
HANA	HANA Compute Units (Disk)	hana_hcu_used_count_5m (type: disk)	Used quota for disk space consumption during the last 5 minutes  A quota represents the available system size and therefore the maximum allowed consumption of a resource. Each resource is measured against the quota independently. For SAP HANA database resources, the quota is measured in HANA compute units (HCUs).

Area	Metric Label	Technical Metric Name	Description
HANA	HANA Compute Units (Memory)	hana_hcu_used_count_5m (type: memory)	Used quota for memory utilization during the last 5 minutes  A quota represents the available system size and therefore the maximum allowed consumption of a resource. Each resource is measured against the quota independently. For SAP HANA database resources, the quota is measured in HANA compute units (HCUs).
Application Jobs	Average Application Job Delay	abap_system_appl_jobs_avg_delay_s_5m	Average application job delay in seconds during the last 5 minutes
Application Jobs	Maximum Application Job Delay	abap_system_appl_jobs_max_delay_s_5m	Maximum application job delay in seconds during the last 5 minutes
Application Jobs	Delayed Application Jobs	abap_system_appl_jobs_delayed_count_5m	The number of delayed application jobs during the last 5 minutes.  An application job is counted as delayed when its delay exceeds a certain threshold. The default configuration uses a threshold of 60 seconds.
Application Jobs	Failed Application Jobs	abap_system_appl_jobs_failed_count_5m	The number of failed application jobs during the last 5 minutes
Application Jobs	Running Application Jobs	abap_system_appl_jobs_running_count	The number of currently running application jobs
Application Jobs	Finished Application Jobs	abap_system_appl_jobs_success_count_5m	The number of successfully finished application jobs during the last 5 minutes
Application Jobs	Application Logs	abap_system_appl_logs_count	The total number of application logs
Application Jobs	Application Logs with Errors	abap_system_appl_logs_errors_count_5m	The number of application logs with errors during the last 5 minutes

## Setup of ABAP Environment Monitoring with SAP Cloud ALM

For more information about setting up ABAP environment monitoring with SAP Cloud ALM, see the tutorial [Monitor An SAP BTP ABAP Environment Service Using SAP Cloud ALM](#) and the documentation [Setup for SAP BTP ABAP Environment](#).

### More Information

[SAP Focused Run on SAP Support Portal](#)

[Health Monitoring in SAP Focused Run](#)

[SAP Cloud ALM on SAP Support Portal](#)

[Health Monitoring in SAP Cloud ALM](#)

[Monitoring the Health of the ABAP System in the Cloud](#) (blog post on SAP Community)

### 6.3.3.2.2 Local Health Monitoring

With the [Health Monitoring](#) app, you can check the health of your local ABAP system.

#### Context

This app allows you to monitor your system health **locally** directly on your ABAP system.

You get an overview of the metrics for the monitored system on one overview page. The health metrics are represented on individual cards with a KPI that summarizes the current status and with a history of the metrics to, for example, compare trends, identify performance bottlenecks, or investigate resource shortages.

The [Health Monitoring](#) app offers the following features:

- The time filter settings apply globally to all cards on the screen, that is, to all KPIs and charts.
- All charts are displayed on one screen and you can easily compare metrics over time.
- Multiple characteristics are visible in one chart if applicable.
- From some cards, you can navigate to related apps for further analysis or issue fixing. For more information, see the in-app help for the individual cards.

#### ⓘ Note

If you want to monitor the health of multiple systems **centrally**, you can use SAP Cloud ALM. For more information, see [Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#).

## Available Metrics for the ABAP Environment

In the [Health Monitoring](#) app, the following metrics for application checks are available for the ABAP environment:

Metrics for the ABAP Environment in Health Monitoring

Metric Card	Technical Metric Name	Description
<a href="#">ABAP Compute Units</a>	abap_acu_used_count_5m	<p>Used quota for ABAP system resources and resources that were allocated during the last 5 minutes</p> <p>A quota represents the available system size and therefore the maximum allowed consumption of a resource. Each resource (memory or CPU) is measured against the quota independently. For ABAP system resources, the quota is measured in ABAP compute units (ACUs).</p> <p>The allocation of ABAP system resources (memory and CPU) is also measured in ABAP compute units (ACUs). In a static service plan, the allocation corresponds to the quota limit of the service plan. In an elastic service plan, the allocation of resources is handled in an adaptive way.</p> <p>The number of ACUs on this card refers to the resource (memory or CPU) with the highest quota usage.</p>
<a href="#">ABAP Runtime Errors</a>	abap_core_dump_count_5m	The number of ABAP runtime errors during the last 5 minutes

Metric Card	Technical Metric Name	Description
<a href="#">Application Jobs</a>	<ul style="list-style-type: none"> <li>Delayed application jobs: abap_system_appl_jobs_delayed_count_5m</li> <li>Failed application jobs: abap_system_appl_jobs_failed_count_5m</li> <li>Running application jobs: abap_system_appl_jobs_running_count</li> <li>Successful application jobs: abap_system_appl_jobs_successful_count_5m</li> </ul>	<p>The number of application jobs, which includes the following:</p> <ul style="list-style-type: none"> <li>The number of delayed application jobs during the last 5 minutes. An application job is counted as delayed when its delay exceeds a certain threshold. The default configuration uses a threshold of 60 seconds.</li> <li>The number of failed application jobs during the last 5 minutes</li> <li>The number of currently running application jobs</li> <li>The number of successfully finished application jobs during the last 5 minutes</li> </ul> <p>The KPI on this card refers to the number of recently failed application jobs. You can compare this value with the successfully finished application jobs to get an overview of your overall application job health.</p>
<a href="#">Application Logs</a>	abap_system_appl_logs_count	The total number of application logs
<a href="#">Application Logs with Errors</a>	abap_system_appl_logs_errors_count_5m	The number of application logs with errors during the last 5 minutes
<a href="#">Average Application Job Delay</a>	abap_system_appl_jobs_avg_delay_s_5m	Average application job delay in seconds during the last 5 minutes
<a href="#">Captured ABAP Statistics Records</a>	abap_system_ksr_captured_count_5m	<p>The number of ABAP statistics records that were captured by the <a href="#">Capture Request Statistics</a> app during the last 5 minutes</p> <p>In the <a href="#">Capture Request Statistics</a> app, you can select the checkbox <a href="#">Health Monitoring</a> for any capture profile that you create. The KPI on this card is the total number of captured ABAP statistics records during the last 5 minutes for profiles with the checkbox <a href="#">Health Monitoring</a> selected.</p> <p>For more information about one use case of this metric, see <a href="#">Monitoring Expensive Outbound Communication [page 2940]</a>.</p>

Metric Card	Technical Metric Name	Description
<i>Critical Number Range Intervals</i>	<code>abap_system_nr_critical_intervals_pct</code>	<p>Fill ratio of number range intervals</p> <p>Only critical number range intervals are shown. When you create a number range interval, you can define the critical remaining fill ratio when a warning is generated.</p> <p>If a critical number range was reported and is extended, it's reported once again with its now uncritical, good fill ratio to report the fixed state. After that, this number range is dropped from reported number range intervals because it's not critical anymore.</p>
<i>Current Unique Users</i>	<ul style="list-style-type: none"> <li>Sessions: <code>abap_system_sessions_count</code></li> <li>Users: <code>abap_system_users_count</code></li> </ul>	The number of current unique users and sessions in the ABAP system
<i>E-Mail Transmissions (Last Day)</i>	<code>abap_system_email_error_count_1d</code>	The total number of e-mails and the number of e-mails with errors during the last day
<i>E-Mail Transmissions (Last 5 Minutes)</i>	<code>abap_system_email_error_count_5m</code>	The total number of e-mails and the number of e-mails with errors during the last 5 minutes
<i>Expiry of Client Certificates</i>	<code>abap_system_client_cert_expiry_d</code>	<p>Expiry of client certificates in days</p> <p>With this metric, you can monitor whether any client certificates expire that you have uploaded to the ABAP system for the communication with systems or services outside the ABAP system, for example, BTP services. Make sure that your client certificates are valid, so that communication doesn't break off.</p> <p>If the client certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>

Metric Card	Technical Metric Name	Description
<i>Expiry of Communication System Certificates</i>	abap_system_comsys_cert_expiry_d	<p>Expiry of communication system certificates in days</p> <p>With this metric, you can monitor whether any communication system certificates uploaded to the ABAP system expire for the communication with systems or services outside the ABAP system. Make sure that your communication system certificates are valid, so that communication doesn't break off.</p> <p>If the communication system certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>
<i>Expiry of Trust List Certificates</i>	abap_system_trusts_cert_expiry_d	<p>Expiry of trust list certificates in days</p> <p>With this metric, you can monitor certificates from communication partners that you have added to the certificate trust list in your ABAP system. The metric shows whether any of these trusted certificates expire for the communication with systems or services outside the ABAP system. Make sure that your trusted certificates are valid, so that communication doesn't break off.</p> <p>If the trusted certificate expires within 7 days, it's marked red. If it expires within 30 days, it's marked yellow.</p>
<i>HANA Compute Units (CPU)</i>	hana_hcu_used_count_5m	<p>Used quota for SAP HANA database resources during the last 5 minutes</p> <p>A quota represents the available system size and therefore the maximum allowed consumption of a resource. Each resource (memory, disk or CPU) is measured against the quota independently. For SAP HANA database resources, the quota is measured in HANA compute units (HCUs).</p> <p>The number of HCUs on this card refers to the resource (memory, disk or CPU) with the highest quota usage.</p>

Metric Card	Technical Metric Name	Description
<i>HANA Out-of-Memory Events</i>	hana_db_oom_event_count_5m	The number of out-of-memory events on the SAP HANA index server during the last 5 minutes
<i>Locked Business Users</i>	abap_system_locked_bus_usr_count	The number of locked business users
<i>Locked Communication Users</i>	abap_system_locked_com_usr_count	The number of locked communication users
<i>Maximum Application Job Delay</i>	abap_system_appl_jobs_max_delay_s_5m	Maximum application job delay in seconds during the last 5 minutes
<i>Table Change Deltas</i>	abap_system_bc_dbatalog_delta_count_1d	Table changes during the last day per database table for database tables with more than 10,000 changes
<i>Table Changes</i>	abap_system_bc_dbatalog_count_1d	Total table changes during the last day per database table for database tables with more than 50,000 changes

### 6.3.3.2.3 Health Monitoring Using Your Own Metrics

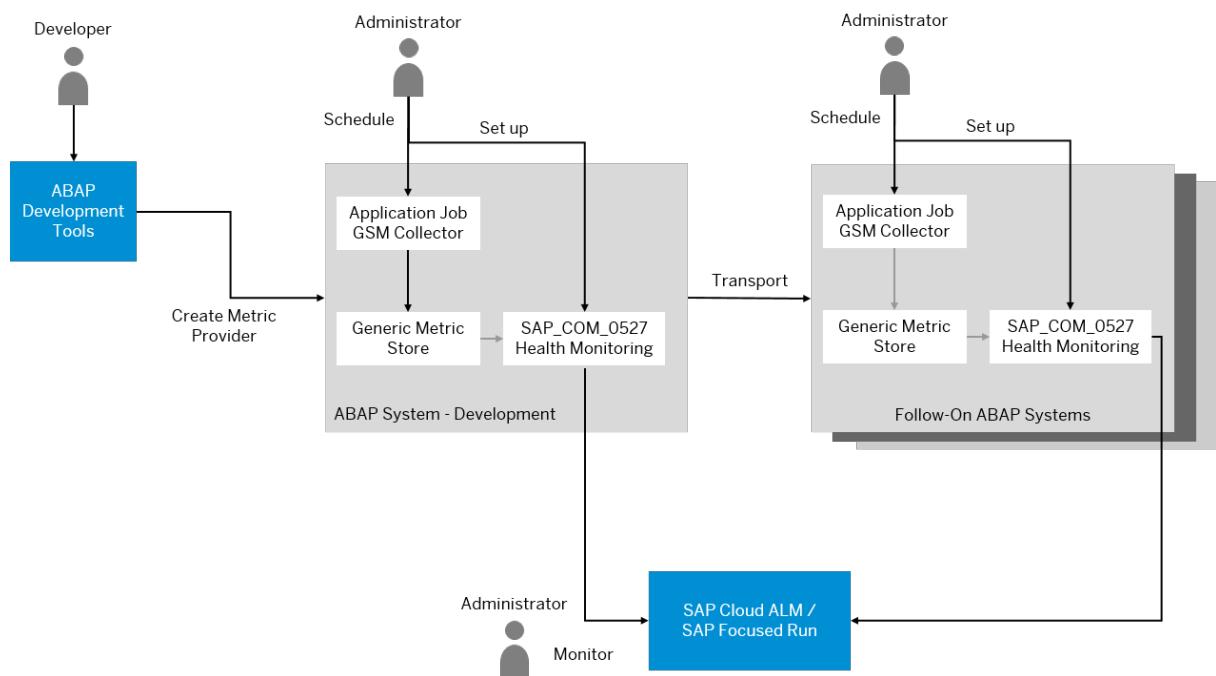
In addition to using the metrics for health monitoring provided by SAP, you can define your own application-specific metrics for the health monitoring of your customer and partner solutions running on the ABAP environment. The new metrics can be consumed by Health Monitoring in SAP Cloud ALM.

#### Background and Use Case

As a customer or partner, you can use SAP Cloud ALM or SAP Focused Run to monitor the health of your systems. As part of the health monitoring in SAP Cloud ALM and SAP Focused Run, you get metrics defined by SAP that support you to watch technical issues such as the number of ABAP runtime errors, failed application jobs, or locked business users, for example, in the ABAP environment.

For your own customer or partner solutions running on the ABAP environment, you might want to define your own application-specific metrics to be able to monitor the health of these solutions. As a partner, you might also want to provide additional health monitoring metrics for your partner solution to your customers.

## Architecture Overview



To define, set up, and use your own metrics, you proceed as follows:

1. Using ABAP Development Tools (ADT), a developer at your customer or partner company defines the relevant metrics for your solution.
2. As an administrator, you schedule an application job using the *Schedule Metric Provider Collection* app. The application job collects the metrics that you have defined and stores the values in the generic metric store.

### ⓘ Note

Make sure that you schedule this job for the development system and all follow-up systems.

3. As an administrator, you define a communication arrangement based on the communication scenario SAP\_COM\_0527 (*Health Monitoring*) in your ABAP system.  
As a result, a technical background job pushes the measured metrics automatically to SAP Cloud ALM or SAP Focused Run.

## Related Information

[Central Health Monitoring Using SAP Focused Run and SAP Cloud ALM \[page 2956\]](#)

[Developing Metrics for Health Monitoring \[page 1189\]](#)

### 6.3.3.3 Displaying and Analyzing ABAP Runtime Errors

With the [ABAP Runtime Errors](#) app, you can display and analyze the ABAP runtime errors that have occurred in your current tenant.

#### Displaying ABAP Runtime Errors

When you open the app, you get a list of all ABAP runtime errors that have occurred in your current tenant and that have written a short dump.

You can search this list for certain dates and times, users, runtime errors, exceptions, and programs. In addition, you can filter the content of the list by using the corresponding filters.

##### Note

You can store the full content of an ABAP runtime error in a `.zip` file on your local drive by choosing the [Download](#) button.

#### Analyzing ABAP Runtime Errors

To analyze an ABAP runtime error, click on a row in the list. This takes you to detailed information about this runtime error:

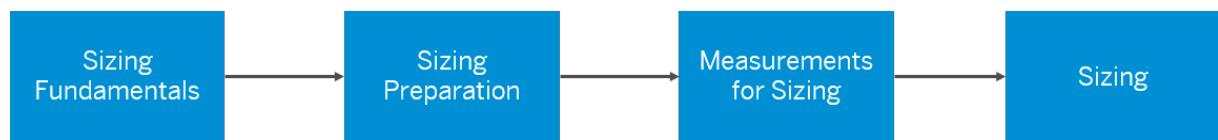
- The [General](#) section shows a description of what happened, an analysis of the error and information on where the program terminated.
- The [Source Code Extract](#) section shows an extract of the ABAP source code where the error occurred. You can work in the source code now and return to the source code position where the error occurred at any time by choosing [Go to Line with Runtime Error](#). You can now choose whether you want to go to the ABAP code in the ABAP development tools for Eclipse (ADT) to debug or change the development object or whether you want to display the code read-only in HTML.
- The [Active Calls/Events](#) section lists the call stack with information on where the active calls and events occurred. From here, you can also navigate to the ABAP code in the ABAP development tools for Eclipse (ADT) or display it in HTML.
- Some HTTP requests from applications can result in failures that are recorded as ABAP runtime errors. For these cases you can find more information about the application call in the [Application Information](#) section.

## 6.3.3.4 ABAP System Sizing

This documentation describes the principles of how to perform system sizing for a custom application operated in the ABAP environment.

### About this Documentation

This documentation introduces you to the following:



- [Sizing Fundamentals \[page 2972\]](#)
  - [Sizing Preparation \[page 2976\]](#)
  - [Measurements for Sizing \[page 2979\]](#)
  - [Sizing \[page 2983\]](#)
1. **Sizing fundamentals**  
You learn more about sizing occasions and methods, especially about expert sizing and how sizing works for scalable applications in the ABAP environment.
  2. **Sizing preparation**  
Important preparatory steps are to identify the business processes that are relevant for sizing and to define test cases for them.
  3. **Measurements for sizing**  
You can perform measurements for sizing using the *Capture Request Statistics* app.
  4. **Sizing**  
You can use the *Perform System Sizing* app to perform the actual sizing of your ABAP system based on what you have measured before.

As the types of applications developed and operated in the ABAP environment are highly diverse, use this documentation as a rough guideline. This documentation assumes that in your business scenario, most business processes comprise the synchronous processing of single transactions. These business processes are the basis for measuring and calculating sizing values. If your business application mainly comprises other types of processing, for example, the mass processing of background jobs at specified dates and times, you need other sizing methods to achieve meaningful results.

This documentation focuses on the sizing of the ABAP layer. It doesn't include sizing for the SAP HANA database.

## 6.3.3.4.1 Sizing Fundamentals

Sizing describes the process of translating business requirements into hardware requirements.

Sizing determines resource requirements such as CPU processing power, physical memory, as well as disk and network bandwidth. The outcome is influenced by both business and technological aspects. This means that the number of users and their application usage is as important as the behavior of the application and the underlying technology stack.

### 6.3.3.4.1.1 Sizing Occasions and Methods

Sizing makes sense at different occasions, such as before a go-live or after an increased usage of applications. You can use different sizing methods, ranging from educated guess to expert sizing.

#### Occasions for Sizing

Sizing can be performed at different occasions:

- Before the initial go-live of an application (greenfield sizing)
- After the initial go-live and after an increased usage of applications, or after functionality has been added or changed (delta sizing or resizing)

Reliable sizing can only be achieved if the hardware as well as the platform layer used are scalable, but most importantly, the application software must be scalable. Load tests are used to check the scalability of all involved components.

In a typical productive environment, 80% of the workload is caused by 20% of the business processes. Consequently, detailed sizing focuses on these relevant business processes. Sizing for the remaining business processes is typically done using an educated guess.

As a rule of thumb, sizing should target an average resource utilization of no more than 70%. Keep extra space available for peak times as well as for anticipated future growth because of increasing business activity.

#### → Recommendation

We recommend that you consider a resizing of the environment if the resource utilization frequently exceeds 80%. Going beyond 90% resource utilization in normal operation mode usually causes resource contention in individual components and can considerably impact user experience with longer response times.

#### Sizing Methods

There are different sizing methods that allow you to perform sizing with different efforts required, but also resulting in different levels of accuracy. Sophisticated sizing methods are applied in environments where

hardware resources are the main cost driver. Simple and less elaborate sizing methods can be applied in smaller environments, or to get a starting point for less critical scenarios.

Sizing Method	Advantages	Disadvantages
Educated guess (Rule of thumb)	<ul style="list-style-type: none"><li>• Easy to apply</li><li>• Delivers a quick rough estimate</li></ul>	<ul style="list-style-type: none"><li>• Depends on many assumptions</li><li>• Highly depends on the experience of the person who makes the guess</li></ul>
T-shirt sizing (Predefined application model and categories of workload sizes)	<ul style="list-style-type: none"><li>• Easy to apply</li><li>• Easy to understand</li></ul>	<ul style="list-style-type: none"><li>• Depends on many assumptions</li><li>• Requires an underlying sizing model that fits well to the actual application</li></ul>
Expert sizing (Manual measurement and creation of sizing formula)	<ul style="list-style-type: none"><li>• Delivers the most realistic results</li><li>• Allows for more variables</li><li>• Transparent sizing model</li></ul>	<ul style="list-style-type: none"><li>• Suggests accuracy that sizing can't deliver</li><li>• Requires the highest efforts</li></ul>

### 6.3.3.4.1.2 Expert Sizing

As the ABAP environment is a platform-as-a-service offering, sizing in this context refers to applications that aren't standard SAP coding but are custom development of either an SAP partner or an SAP customer. This fact usually prevents the usage of the educated guess or t-shirt sizing method, as there's typically no reference available. This documentation therefore focuses on the expert sizing method.

Expert sizing involves the following steps:

1. Identify the sizing-relevant (main) business scenarios and their variable dimensions.
2. Determine the throughput requirements during defined time periods and their day-time distribution. As part of the requirement determination, separate scenarios that are time-critical from those scenarios that can be postponed to times with lower load.
3. Define test cases for each relevant scenario with a reasonable number of input parameters and assumptions. These test cases must be self-contained and idempotent, which means repeatable with a stable load profile.
4. Execute each test case in single-user mode to measure its individual workload profile.
5. Create sizing models based on the results of the previous steps.
6. Create a structured load analysis to verify and refine the sizing model regarding scalability aspects (mass user testing).

#### ⓘ Note

This step isn't in the scope of this document.

7. To simplify the consumption of the sizing formulas resulting from expert sizing, you can also define and precalculate representative t-shirt sizes for the system.

#### → Recommendation

Changes to the underlying technology stack (for example, configuration changes in the ABAP environment), as well as changes in the application behavior (for example, enhanced functionality in a

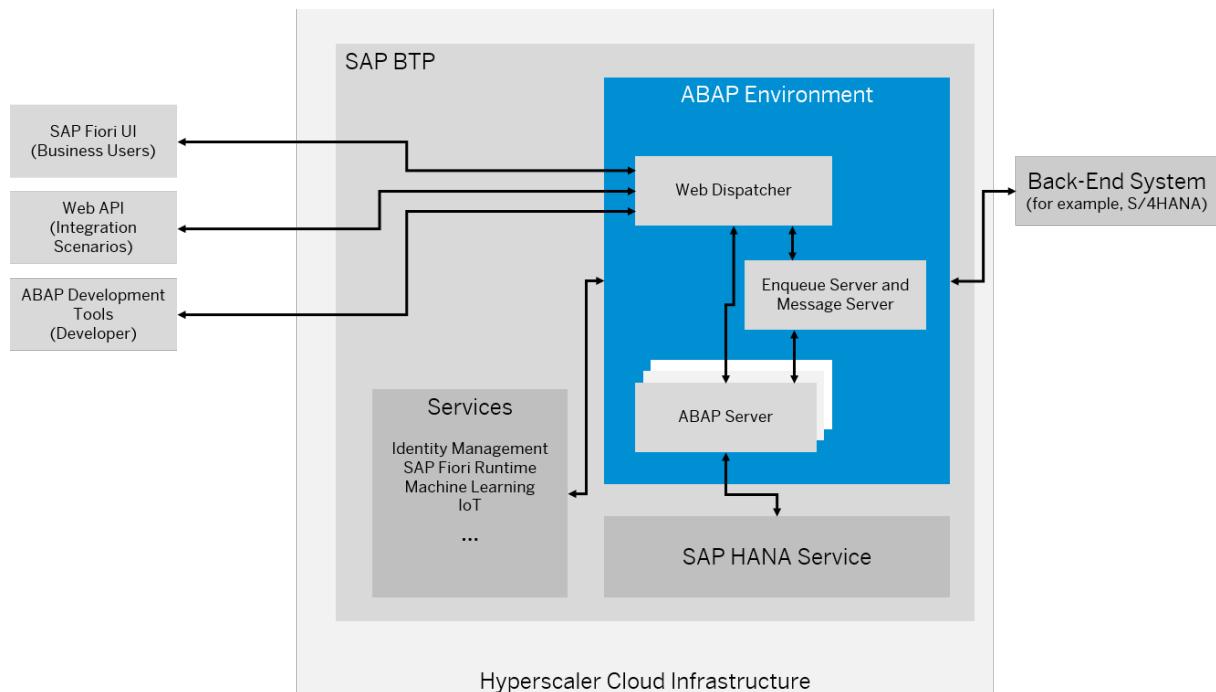
(new software release) can invalidate your sizing results. We recommend that you validate from time to time whether the sizing is still adequate for your system workload.

### 6.3.3.4.1.3 Scalable Applications in the ABAP Environment

Get an overview of the technical architecture of an ABAP environment. It's important to understand which aspects are conditioned by this architecture and are therefore relevant for creating scalable applications in the ABAP environment and for the sizing of such applications.

#### Architecture of the ABAP Environment

The following figure provides an overview of the technical architecture of the ABAP environment:



Both the developer and the business user access the ABAP environment using HTTPS OData calls. The web dispatcher of the ABAP system dispatches these calls to an ABAP server. The respective ABAP server processes the request and might access the SAP HANA database to read or modify data. Besides user-driven calls, the web API can be used to integrate ABAP environment-based applications with other systems, for example, to consume IoT data.

While there's only one web dispatcher and one enqueue and message server, the number of ABAP servers can be increased based on demand (scale-out approach). There are always at least two ABAP servers.

The ABAP environment itself consumes reuse services of SAP BTP, for example, identity management or the SAP Fiori UI runtime. The ABAP environment developer can consume additional reuse services such as machine learning. Furthermore, predefined connectivity services of SAP BTP can be consumed to connect an ABAP environment with an SAP S/4HANA (or SAP ERP) back-end system.

## Relevance for Sizing

The following aspects are important regarding the scalability of the ABAP environment-based applications:

- The ABAP environment is designed to scale primarily via the number of ABAP servers (scale-out), and not via the size of individual ABAP servers (scale-up). It's therefore important to split larger work packages into smaller ones, and to process these smaller packages in parallel on many rather small ABAP servers .
- With the in-memory architecture of SAP HANA and its support for heavily parallelized operations, it has become popular to push down application logic into the database layer (ABAP supports this pushdown using ABAP-Managed Database Procedures). While this pushdown remains a valid use case for operations that are time-critical or benefit greatly from being executed close to the data, care must be taken not to overload the central (and expensive) SAP HANA instance. In many cases, scaling out by using more (comparatively cheap) ABAP servers provides a better total cost of ownership.
- The ABAP environment and SAP HANA as a cloud service are co-located in a single virtual data center (one hyperscaler region, but not necessarily the same data center). Therefore, the network latency between these components is still slightly higher than in a typical on-premise environment. Consequently, performance anti-patterns such as small `SELECT` queries within loops are even more problematic in an ABAP environment than in an on-premise setup.
- As for any cloud service, network latency is also a limiting factor for clients interacting with the ABAP environment. Avoid performing many small calls to the ABAP environment.

Keep these aspects in mind when you plan scalable applications in the ABAP environment and their sizing.

### 6.3.3.4.1.4 Native Storage Extension for the ABAP Environment

You can use SAP HANA Native Storage Extension for selected database tables to save in-memory storage.

#### SAP HANA Native Storage Extension

SAP HANA Native Storage Extension (NSE) is a general-purpose, built-in warm data store in SAP HANA that lets you manage less-frequently accessed data without fully loading it into memory. It integrates disk-based or flash-drive based database technology with the SAP HANA in-memory database for an improved price-performance ratio.

For more information, see [SAP HANA Storage Extension](#) in the SAP HANA administration guide.

#### Enabling Native Storage Extension for Selected Database Tables

When you create or alter database tables in the ABAP Development Tools, go to the technical table settings. In the *Load Unit* field, select *Page Enforced*.

## 6.3.3.4.2 Sizing Preparation

Before you start any measurements for system sizing, it's important that you prepare system sizing carefully.

Such preparations include first and foremost identifying the main business processes that use the custom application for which you want to perform sizing. As a second step, formulate test cases that you want to use for the actual measurements.

### 6.3.3.4.2.1 Defining the Sizing-Relevant Business Processes

Before the actual sizing, identify sizing-relevant business processes and determine the throughput requirements for each of these business processes.

#### Context

##### ⓘ Note

Examples used in this sizing documentation refer to the ABAP flight reference scenario of the ABAP RESTful Programming Model. For more information, see [Downloading the ABAP Flight Reference Scenario](#).

#### Procedure

1. Identify the main business processes of the application as well as their variable dimensions.

The main business processes included processes that are critical for the company, for example, from revenue, reputation, or legal perspective.

Examples of variable dimensions are the number of items in a sales order, or the number and size of documents uploaded by a user. It's unavoidable to make some assumption on these variables to limit the complexity of the resulting sizing model. As an example, it can be sufficient to assume an average of 10 items per sales order, instead of making items per sales order an input parameter of the sizing model.

2. Determine throughput requirements for each sizing-relevant business process, with its distribution over a typical business day. Determining throughput requirements also includes the separation of business processes that are time-critical from those processes that can be postponed to times of low load. For example, a background job performing some reporting activity can be shifted into non-business hours.

## Results

The resulting overview of sizing-relevant business processes can look as follows, for example:

Flight Reference Example: Sizing-Relevant Business Processes

Business Process	Load Type	Process Executions per Minute	Criticality	Variable Dimensions
Creation of a new travel request	HTTPS / User-driven	160 per minute during main business hours (8 am to 7 pm)	High	Number of bookings per travel
Approval of a travel request	HTTPS / User-driven	160 per minute during main business hours (8 am to 7 pm)	High	--
Calculation and sending of metrics for reporting	Batch Job / Scheduled	Once a day	Low	Number of travel requests created per day

In this example, variable dimensions have been turned into assumptions to simplify the sizing model and to reduce complexity of the overall sizing procedure. For the flight reference example, we can assume an average of 2 bookings per travel (inbound and outbound flight). With eliminating this variable dimension, the only remaining input parameter of the respective business process is the number of process executions expected per minute.

### 6.3.3.4.2.2 Defining Test Cases for Sizing

After you've identified the sizing-relevant business processes, define test cases for each business process for baseline measurements.

It's important that these test cases are self-contained and idempotent, which means they're repeatable and return the same resource requirements after each repetition. Such a careful design of test cases is important to avoid side effects because, for example, data volume processed by SAP HANA increases during query execution (empty table versus table with high volume of irrelevant data).

Furthermore, make sure that each test case is limited to a single business process. For example, split the creation and the approval of a travel request into two test cases because creation and approval are performed by different users and at different points in time.

### Example

A test case description for the business process *Creating a new travel request* can look as follows:

Test Case Example: Travel Request

Step ID	Description	Input Fields	Input Values
<b>Preparation Steps (Not Measured)</b>			

Step ID	Description	Input Fields	Input Values
SystemLogon	Link to SAP Fiori launchpad	User Password	<tester name>, <tester password>
<b>Test Case (Measured)</b>			
OpenApp	Open the travel management app from the launchpad (or using the Fiori Elements preview).	--	--
CreateTravel_1a	Choose <i>Create</i> from the table toolbar of the travel overview.	--	--
CreateTravel_1b	1. Open the search help of field <i>Customer ID</i> . 2. Choose <i>Go</i> without entering any search data. 3. Select an arbitrary entry from the persons listed.	Customer ID	<arbitrary, using search help>
CreateTravel_1c	1. Open the search help of field <i>Agency ID</i> . 2. Choose <i>Go</i> without entering any search data. 3. Select an arbitrary entry from the agencies listed.	Agency ID	<arbitrary, using search help>
CreateTravel_1d	1. Open the search help of the currency field next to <i>Total Price</i> . 2. In the <i>Currency</i> field, enter <b>EUR</b> and choose <i>Go</i> . 3. Select <b>EUR</b> .	Currency	EUR
CreateTravel_1e	Enter the remaining input and choose <i>Save</i> .	Starting date End date Total price Booking fee Status Description	<arbitrary> <arbitrary> 0 9.99 0 <optional>
<b>Cleanup Steps (Not Measured)</b>			
DeleteTravel	Delete the created travel from either the travel overview or the travel details.	--	--

### 6.3.3.4.3 Measurements for Sizing

After you've defined test cases for sizing-relevant business processes, you can now measure the workload that each business process generates.

#### ⓘ Note

This section of the documentation describes the steps necessary to prepare and execute a single test case for a business process. To obtain a combined sizing of all relevant business processes (see [Defining the Sizing-Relevant Business Processes \[page 2976\]](#)), perform the steps described in this section for each business process individually and add up the results.

#### 6.3.3.4.3.1 Measurement Prerequisites

To ensure meaningful and reproducible results, make sure that prerequisites for the test system and test organization are met before measurements for sizing are taken.

Before you start testing, check the following:

- The setup and configuration of the test environment should be equivalent to a production environment. As you only perform single user tests, the test environment doesn't need the same hardware resources, but make sure that the following is the same as in a production system:
  - Software releases of all software components involved in the stack
  - Settings of the application that influence its behavior, for example, Customizing settings that control internal data verification steps
- Moreover, prevent any concurrent workload that consumes a noticeable amount of resources during the test. Having some house-keeping jobs active in the background is unproblematic. You should, however, avoid that tests compete with other, resource-intensive business processes.
- The test environment should contain a defined amount of existing business data. The SAP HANA database can consume different amounts of resources in a system with no business data compared to a system with a significant amount of existing business data.
- All test cases should be executed with a specific test user. A dedicated test user makes it much easier to separate calls originating from the test case execution from other activities, for example, calls originating from measurement tools. This test user should have the same authorizations that a corresponding business user has in the relevant business process.
- Consider caching effects of the application. The initial load of an application can consume more resources than a later use of the same application. To ensure a measurement that considers caching effects, execute the business scenario once without capturing the workload and a second time with capturing it.

## 6.3.3.4.3.2 Defining a Capture Profile for Sizing

With the [Capture Request Statistics](#) app, you can capture request statistics to find out which activities are running in your ABAP system. For sizing, you can define a capture profile that records defined activities of a business user or business process.

### Prerequisites

You've defined one or multiple test cases to measure the workload of sizing-relevant business processes (see [Defining Test Cases for Sizing \[page 2977\]](#)).

### Context

With the definition and activation of a capture profile, you can capture system activities. Requests are recorded with technical information, such as response time, program name, or CPU time. The request statistics serve as input for ABAP system sizing using the [Perform System Sizing](#) app (see [Performing System Sizing \[page 2983\]](#)).

#### ⓘ Note

In this documentation, the assumption is that workload for a particular business process is measured by capturing the request statistics generated by the activities of a business user who acts as a tester. With such a measurement approach, you define the profile that contains the technical user name of the tester and a time period reserved for testing.

### Procedure

1. In the SAP Fiori launchpad of the ABAP environment, under [Technical Monitoring](#), choose the [Capture Request Statistics](#) tile.
2. Choose [Create](#).
3. Enter the following data:

Field	What to enter
<a href="#">Profile ID</a>	Enter a short, meaningful name for your profile. Don't use the prefix <code>SAP_</code> because this prefix is reserved for profiles predefined by SAP.
<a href="#">Description</a>	If needed, you can enter a longer description of your profile.

Field	What to enter
<i>Start/End Time</i>	The start and end time specify when request statistics were captured by the app. These fields are automatically filled after you have activated the profile.
<i>Retention Time</i>	Select how long you want to keep the captured request statistics stored in the system, for example, 14 days.
<i>Record Limit</i>	<p>Enter the maximum number of records that you want the system to capture per minute, for example, 1,000 records.</p> <p>The record limit prevents that too much database memory is consumed by captured request statistics. The maximum possible record limit per minute is 1,000.</p>
<div style="border: 1px solid #ccc; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>To get a realistic idea of the workload, consider a relatively high record limit. However, you must keep in mind that you generate load on your system while request statistics are captured.</p> </div>	
<i>Sampling Rate</i>	Enter a sampling rate, which is the probability of capturing one single ABAP statistics record in percent.
<div style="border: 1px solid #ccc; padding: 10px;"> <p><b> ⓘ Note</b></p> <p>To get a realistic idea of the workload, consider a relatively high sampling rate, if possible 100. However, you must keep in mind that you generate load on your system while request statistics are captured.</p> </div>	
<i>Target User Group</i>	Choose one of the target groups starting with <i>Customer</i> .
<i>Profile Type</i>	This field is automatically populated by the system. Only static profile types (with fixed values) are possible.
<i>Profile Owner</i>	The profile owner is automatically set to <i>Customer</i> when you create a new profile.

4. Choose *Create*.

The profile has been created, but is still inactive and has no filter conditions yet.

5. To define the filter conditions under which request statistics are captured, choose *Create*.

6. With *Field ID*, *Operator*, *High*, and *Low*, you can select a filter condition that determines what kind of data is captured.

For example, if you want to capture all request statistics relating to your tester, select *User ID* as field ID, *include equal to* as operator and the technical user name of the tester.

7. Choose *Create*.

8. To enable capturing request statistics using the profile, choose *Activate*.

If you use the capturing of request statistics to measure sizing, enter the planned start and end time of your test case.

### 6.3.3.4.3.3 Executing the Test Cases

Instruct your testers to perform the steps described in the test cases to measure the workload for sizing.

#### Prerequisites

Before testing starts, make sure that the following prerequisites are met:

- Testers have the authorizations that correspond to that of a business user.
- A capture profile exists, and it is set to *Active* (see also [Defining a Capture Profile for Sizing \[page 2980\]](#)).
- Testers perform their tests after the start time and before the end time defined in the capture profile.
- When you define and activate a profile, the start time of the profile must be later than or equal to now. The testers can start testing immediately. However, note that it takes up to two minutes before the records are processed by a collector job that runs in the background. Therefore, it can take up to two minutes until the first request statistics from your test are shown in the sizing app. In addition, it can take up to two minutes after the defined end time of the profile until all records have been processed and are shown in the sizing app.

#### Context

The testers' system activities are captured to estimate the workload as a basis for system sizing.

#### Procedure

1. Instruct your testers to execute the preparation steps as documented in [Defining Test Cases for Sizing \[page 2977\]](#).
2. Instruct your testers to execute the steps of the test case within the defined time period for testing.
3. After testing, wait for two minutes before you set the profile to *Inactive* again.

##### ⓘ Note

If a profile is set to *Inactive* before the collector of the request statistics runs another time, all ABAP statistics records get lost that would have been captured with the last collector run.

#### Results

The ABAP statistics records relating to the tester activities in the ABAP system have been captured. To ensure stable results, consider defining multiple capture profiles with different test times for one test case and repeat the measurement 2-3 times.

## 6.3.3.4.4 Sizing

After you have prepared measurements for sizing and a tester has executed the relevant test cases, you can now perform system sizing for each relevant business process.

With the help of the [Perform System Sizing](#) app, you can get sizing results for individual test cases in ABAP compute units. If you combine your results for all business processes, you can get a sizing for the overall business scenario.

### 6.3.3.4.4.1 ABAP Compute Units

When you order an ABAP environment in SAP BTP, the ABAP system size is specified in ABAP compute units (ACUs). One ABAP compute unit comprises the total ABAP memory usable by applications, the ABAP work process time per minute, and the ABAP CPU time per minute.

When you create an ABAP system, you must decide on the system size in ABAP compute units. One ABAP compute unit represents 16 GB.

The system consists of at least 2 application servers of the size of 8 GB each.

When you order 4 ACUs or more, the system consists of at least 2 application servers of the size of 32 GB each.

The memory limit for a single session is 4 GB, independent of the application server size.

### 6.3.3.4.4.2 Performing System Sizing

With the [Perform System Sizing](#) app, you can calculate the required total ABAP memory, the work process time, and the ABAP CPU time in ABAP compute units for a custom application operated in the ABAP environment. You can use these calculations as a basis for ABAP system sizing.

#### Prerequisites

Before you start a system sizing, you've used the [Capture Request Statistics](#) app to capture a sample of the expected workload of the custom application.

##### Note

In the [Capture Request Statistics](#) app, you must create your own profile for capturing the relevant request statistics. For system sizing, you can't use the predefined SAP capture profiles.

## Context

You want to use the captured request statistics as a basis to perform the system sizing of a production system. The *Perform System Sizing* app uses the captured request statistics to calculate the required total ABAP memory, the work process time per minute, and the ABAP CPU time per minute for the custom application and translates these KPIs into ABAP compute units (ACUs).

## Procedure

1. In the SAP Fiori launchpad, under *Technical Monitoring*, choose the *Capture Request Statistics* app.
2. In the list of capture profiles, check whether the profile that you've defined before to capture the request statistics of the custom application is set to *Finished*.
3. Choose the profile from the list.

The profile with its settings is shown.

4. To navigate to the system sizing, choose the *Perform System Sizing* button.

The *Perform System Sizing* app opens with the captured request statistics of the profile that you've chosen.

When you open the app, all captured ABAP statistics records are selected for system sizing, and a system sizing for 10 parallel process executions is shown. Initially, one ABAP compute unit is entered as available, but you can also enter a higher number of available ABAP compute units.

5. Replace the default value of 10 parallel process executions with the expected volume as assumed in your sizing-relevant business process definition (see [Defining the Sizing-Relevant Business Processes \[page 2976\]](#)).

The process execution is the unit to describe the execution of a specific business process by a single business user.

6. Deselect ABAP statistics records from the table that aren't part of the test case definition (see [Defining Test Cases for Sizing \[page 2977\]](#)).

### ⓘ Note

You can use the *User ID* field to deselect requests from users that you want to exclude from system sizing. By default, the field is hidden, but you can display it by choosing it in the table settings of the *ABAP Statistics Records* table.

7. Check whether the required ABAP compute units for total ABAP memory, work process time, and ABAP CPU time are still below acceptable thresholds.

If total ABAP memory, work process time, and ABAP CPU time consume more than 70% of the available ABAP compute units, a warning sign next to the calculated ABAP compute units is shown. If 100% or more is consumed, an error sign is shown.

8. If there are still warnings or errors, increase the number of available ABAP compute units until all error or warning signs disappear and the calculated ABAP compute units (ACUs) for total ABAP memory, work process time, and ABAP CPU time are displayed with a green OK sign.

### ⓘ Note

If you notice an entry in the captured ABAP statistics records that is responsible for a high workload and considerable resource consumption, don't simply accept a high number of required ABAP

compute units for your business process. Consider reviewing the relevant request for optimization and, after the optimization, repeat the sizing.

## Results

Combine the results for multiple custom applications that run in the production system to get an idea of the required system size. After you have found the appropriate number of ABAP compute units for your business scenario, you can proceed with using this number for requesting an ABAP system for production.

### Related Information

[Combining System Sizing Results \[page 2985\]](#)

#### 6.3.3.4.4.3 Combining System Sizing Results

After performing a system sizing for each business process, combine the results for multiple custom applications that are planned to run in the production system. As a result, you get an idea of its required system size in ABAP compute units (ACUs).

Let's use the flight reference scenario and let's assume you've performed system sizing for its business processes *Create a new travel request*, *Approve a travel request*, and *Calculate and send metrics for reporting*. After performing system sizing for each business process, the results can look as follows:

Flight Reference Scenario: Example Sizing Results

Business Process	ABAP Total Memory	Work Process Time	ABAP CPU Time	Process Executions/Minute
Create a new travel request	0.108 ACUs	0.306 ACUs	0.414 ACUs	180
Approve a travel request	0.072 ACUs	0.162 ACUs	0.234 ACUs	180
Calculate and send metrics for reporting	0.025 ACUs	0.331 ACUs	0.092 ACUs	1

The process execution is the unit to describe the execution of a specific business process by a single business user. In the flight reference scenario, we assume that up to 180 travel requests are created and approved each minute, and metrics are calculated and sent for reporting once a minute.

The workload generated by a business process can vary depending on the daytime. In the flight scenario example, let's assume that travel requests are only created and approved between 8 a.m. and 8 p.m., but the

reporting runs in the evening after 8 p.m. Combining the business processes with the volumes expected at different times of day, you can create a summary such as the following:



Here, the ABAP total memory consumption, the work process time, and the ABAP CPU time for the travel requests and approvals must be combined. They add up to more than 0.2 ACUs for ABAP total memory, almost 0.5 ACUs for work process time, and more than 0.6 ACUs for ABAP CPU time. The reporting runs after 8 p.m. and its sizing is lower than the sum of the figures for travel request and approvals. Consequently, the limiting resource for the overall business scenario is the ABAP CPU time, with a maximum requirement of 0.648 ACUs. This ABAP CPU time still fits into the smallest available service plan of the ABAP environment (while also reserving some space for smaller business processes considered irrelevant for sizing).

To simplify the consumption of the sizing formulas that you created, you can optionally define some feasible t-shirt sizes and precalculate their resource requirements. Remember, sizing targets a mean resource utilization of no more than 70%:

#### Flight Reference Scenario: T-Shirt Sizes

T-Shirt Size	Description	Required ABAP Compute Units
S	Up to 180 travels created and approved per minute	1
M	Up to 360 travels created and approved per minute	2
L	Up to 540 travels created and approved per minute	3

### 6.3.3.5 Resilience

The resilience of your ABAP Cloud applications and your data provided by a set of SAP BTP, ABAP environment features.

SAP BTP, ABAP environment provides the following set of features to increase the availability and resilience of your ABAP cloud applications and your data:

- Both the infrastructures for ABAP systems and SAP HANA Cloud databases are set up across multiple availability zones.
- The infrastructures offer automatic restarts for critical components of the Application Server for ABAP, the HANA Cloud database, and connectivity components.
- In case of hardware failures, components are automatically rescheduled to other hardware (subject to availability).
- Enqueue locks are stored redundantly.

#### ⓘ Note

SAP BTP ABAP Environment provides protection against denial-of-service attacks via various defense mechanisms. These are continuously improved to keep a decent protection level. However, a full protection against DoS attacks cannot be guaranteed.

### 6.3.3.6 Backups

The ABAP environment relies on backups from SAP HANA Cloud:

- Backups are retained up to 14 days.
- The recovery point objective (RPO), which is the maximum data loss, is no more than 15 minutes.
- Backups are stored encrypted and redundantly in different availability zones.

#### ⓘ Note

A point-in-time recovery requires a coordinated plan between you as a customer and SAP that clarifies the target point-in-time and the target time of execution.

You can initiate a restore via a service request by using component BC-CP-ABA. To perform a restore, please provide the following information:

- The ID (GUID) of the ABAP environment service instance
- The target point-in-time to which the instance should be restored specified in coordinated universal time (UTC)

#### ⚠ Caution

All data stored between the specified time and the time at which the service request is processed by SAP can't be retrieved.

#### Caution

A point-in-time recovery changes data stored in the ABAP environment service database only. Other data will not be changed, which may lead to inconsistencies with communication integration configurations or integrated services.

#### Restriction

In case of SAP-managed software updates that are part of scheduled maintenance, the target point-in-time can only be set after the last software update.

## 6.3.4 Troubleshoot Custom Apps

After developing or releasing an application to a test or productive system, you might need to troubleshoot your application, for example to debug your business logic, display data of your application, and to analyze short dumps as well as logs and traces.

### Prerequisites

As an administrator, assign business role `Application Support Engineer - Development Support` to a business user. With this role, you enable business users to troubleshoot applications. In addition to this business role, you might need to assign the business role of the application to be analyzed to the business user.

### Business Role Template Application Support Engineer - Development Support

Business role template `Application Support Engineer - Development Support` (`BR_APPL_SUP_ENG_DEV_SUP`) contains all the business catalogs that are required for troubleshooting activities, such as:

- Debugging capability
- Running the data preview
- Executing classrun
- Accessing Feed Reader
- SQL Trace Analysis & SQL Explain Analysis capability

### Business Catalogs

Business role template `Application Support Engineer - Development Support` includes the following business catalogs:

Business Catalog	Authorization
Development Support - Data Preview - Business Data (SAP_A4C_BC_DAT_PRV_DFT_PC)	<p>This business catalog enables you to use the data preview in ABAP development tools for Eclipse for objects that are considered to contain business data, such as:</p> <ul style="list-style-type: none"> <li>• CDS views in language version 5</li> <li>• Client-dependent tables in language version 5 with delivery class A and L</li> <li>• Data of CDS views that can be viewed with or without application of the DCL</li> </ul>
Development Support - Data Preview - Cross-Client and Customizing Data (SAP_A4C_BC_DAT_PRV_RDO_PC)	<p>This business catalog enables you to use the data preview in ABAP development tools for Eclipse for objects that are considered to contain customizing and cross-client data, such as:</p> <ul style="list-style-type: none"> <li>• Client-independent tables in language version 5 with delivery class A and L</li> <li>• Tables in language version 5 with delivery class S, E, and W (metadata)</li> <li>• Tables in language version 5 with delivery class C and G (customizing)</li> </ul>
Development - Class Runner Execution (SAP_A4C_BC_DEV_CLA_RUN_PC)	<p>This business catalog enables you to execute class runners in ABAP development tools for Eclipse.</p>
Development - Data Preview - Released Objects (SAP_A4C_BC_DEV_DAT_PRV_PC)	<p>This business catalog enables you to use the data preview in ABAP development tools for Eclipse for objects that are released for language version 5:</p> <ul style="list-style-type: none"> <li>• SAP-delivered CDS views</li> <li>• CDS Views that are released for usage outside of their own software component</li> <li>• Data of CDS views that can be viewed with or without application of the DCL</li> </ul>
Development - Development Objects Display (SAP_A4C_BC_DEV_OBJ_DIS_PC)	<p>This business catalog enables you to sign in to ABAP development tools for Eclipse and view development objects with read-only authorization.</p>
Development - Analysis and Support (SAP_A4C_BC_DEV_SUP_PC)	<p>This business catalog gives you access to troubleshooting tools such as logs, traces, and the debugger.</p> <p>Business user assigned to the business role Development or any kind of source code changes are not permitted. Nevertheless, business users can execute the class runner in a test or productive system, unless this is not restricted by the administrator.</p>

## Connect to the System

As the Application Support Engineer, you have to connect to the ABAP system using ABAP development tools for Eclipse. This allows you to use the troubleshooting functionality and tools.

### Note

Certain analysis tools, such as the SQL Trace Analysis and SQL Explain SAP Fiori app are only accessible in SAP Fiori launchpad.

See [Connect to the ABAP System \[page 708\]](#).

## Troubleshoot Using Debugging

The most common use case for troubleshooting to analyze the implemented business logic is debugging. The following debugging scenarios exist:

- Debug your own user
- Debug another business user
- Debug a communication user

See [ABAP Debugger](#).

### Note

To find the business user and communication user ID for debugging, navigate to the *Debug Properties View* in ABAP development tools for Eclipse and search for business users (**CB\***) and communication users (**CC\***).

### Debug Your Own User

To debug the application with your own Application Support Engineer user, make sure that you have the business application role assigned.

### Debug Another Business User

To debug the application of another business user, you need to change the breakpoint settings in the debug properties view in ABAP development tools for Eclipse to the business user (**CB\***).

### Debug a Communication User

To debug the application of the communication user, you need to change the breakpoint settings in the debug properties view in ABAP development tools for Eclipse to the communication user (**CC\***).

## Related Information

[Troubleshooting Tools \[page 2991\]](#)

### **6.3.4.1 Troubleshooting Tools**

Apart from debugging, you can also use other troubleshooting tools, such as the Feed Reader view in ABAP development tools for Eclipse, ABAP Profiler as well as the SQL Explain and SQL Trace Analysis SAP Fiori apps.

#### **Feed Reader**

ABAP Development Tools provides a dedicated view, the Feed Reader View, for runtime errors (ABAP short dumps) and gateway errors. See [Feed Reader View](#).

#### **ABAP Profiler**

##### **ABAP Trace**

The ABAP Profiler allows you to examine the performance of your implemented business logic by using function modules and global ABAP classes. You can use the ABAP trace, which is the result of the ABAP Profiler, to identify runtime-intensive statements and to follow the hierarchy of program calls. See [Profiling ABAP Code](#).

##### **ABAP Cross Trace**

If you develop OData services or other ABAP functionality related to the ABAP RESTful application programming model , the ABAP Cross Trace gives you insights into the ABAP RESTful Application Programming Model runtime framework. This includes the processing of OData requests, for example in SAP Fiori applications, and contains functionality similar to the payload trace of SAP Gateway trace in SAP GUI.

#### **SQL Trace Analysis SAP Fiori App**

With the SQL Trace Analysis app, you can access and analyze SQL trace records after you have used the SQL trace function of the ABAP performance trace in ABAP development tools for Eclipse. See [SQL Trace Analysis \[page 2622\]](#).

## **6.4 Administration and Operations in the Kyma Environment**

This is the managed offering of SAP BTP, Kyma runtime (based on the open-source project "Kyma"). The administrators of the Kyma environment take care of setting it up and make sure it is ready for developers to work with. Create your Kyma instance to build applications and extensions to SAP and third-party solutions, manage roles, have your Kubernetes objects backed up, and view metrics and logs.

## ⚠ Caution

Do not make changes in any Kyma-managed namespaces, the `<kube-system>` namespace, or to any resources marked with *Do Not Edit*. Changes to those resources could lead to exclusion or disruptions from the agreed SLA. If you decide to make changes despite this, consult support in advance, and proceed with caution.

### [Create a Kyma Instance \[page 2992\]](#)

Set up a Kubernetes cluster with SAP BTP, Kyma runtime and use it to build applications and extensions to your SAP and third-party solutions.

### [Managing Kyma Runtime Using the Provisioning Service API \[page 2994\]](#)

The SAP Cloud Management service (technical name: `cis`) provides the Provisioning Service API to create and manage available environments. Use the Provisioning Service API to automatically manage and access SAP BTP, Kyma runtime.

### [Available Plans in the Kyma Environment \[page 2998\]](#)

Depending on your global account type, you have access to a different plan that specifies the cluster parameters for the Kyma environment.

### [Provisioning and Updating Parameters in the Kyma Environment \[page 3002\]](#)

You can configure the cluster parameters in the Kyma environment.

### [Add and Delete a Kyma Module \[page 3012\]](#)

To use a Kyma module, you must add it first. Use Kyma dashboard or kubectl to do that. If you don't need the module anymore, delete it to save resources.

### [Access Kyma Application Logs \[page 3014\]](#)

Get insights into your applications, microservices, and Functions by viewing the respective logs. To check out real-time logs immediately, use the Kubernetes functionalities - either in Kyma dashboard, or with kubectl.

### [Kyma Environment Backup \[page 3015\]](#)

#### [Kyma Modules' Sizing \[page 3016\]](#)

The following tables show the resource consumption, such as memory, CPU, and storage, of Kyma modules.

#### [Change Storage Size in Kyma \[page 3019\]](#)

If the amount of data for the applications in your Kyma environment grows, you can expand the storage size for your customer data by resizing the respective Persistent Volume Claim (PVC).

## 6.4.1 Create a Kyma Instance

Set up a Kubernetes cluster with SAP BTP, Kyma runtime and use it to build applications and extensions to your SAP and third-party solutions.

## Prerequisites

Your subaccount has entitlements for Kyma runtime configured. For more information, read [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).

## Context

To set up the Kyma environment on your subaccount, you must create an instance of it. You can easily create it from Service Marketplace, following the same wizard steps as for SAP services and applications. You can configure the instance by selecting the region in which you want to create the cluster and the number of virtual machines to run on it.

### ⓘ Note

The legacy method to set up a Kubernetes cluster is to select *Enable Kyma* in the *Kyma Environment* section of your subaccount overview. However, this way offers fewer configuration options, so we recommend the Service Marketplace.

## Procedure

1. In SAP BTP cockpit, navigate to your subaccount.
2. Go to and select *Kyma Environment*.
3. Select *Create* either from actions in the upper-right corner of the tile or from the tile overview page, to which you get by clicking on the tile.
4. Change the instance name or keep the default name.  
You see the default plan assigned to your account. The default plan and its specification can differ depending on your global account type.
5. To move to the plan configuration view, choose *Next*.
6. In the *Parameters* view, specify the required details. You can also configure your instance parameters by changing its region and the maximum or the minimum number of virtual machines to be created for it. You can also configure a custom identity provider.
7. Confirm changes by selecting *Create*.

## Results

You have created a Kyma environment instance.

### → Tip

If you prefer to work in a terminal or want to automate operations using scripts, there's an alternative to the SAP BTP cockpit: You can create the Kyma environment with the SAP BTP command line interface (btp CLI). See [Creating SAP BTP, Kyma runtime via the SAP BTP cli](#).

To manage a Kyma instance automatically, create a Kyma service binding. The binding enables getting a Kyma kubeconfig, which in turn allows for accessing a Kyma cluster, deploying applications, running tests, and deleting the resources in a fully automated way. See [Managing Kyma Runtime Using the Provisioning Service API \[page 2994\]](#).

## Next Steps

- To get the link to Kyma dashboard, navigate to the overview of the instance under [Instances and Subscriptions](#).
- To manage access to the Kyma environment and Kyma dashboard, assign roles as needed.
- To use functionalities such as telemetry and eventing, or to use BTP services, add the respective [Kyma \[page 74\]](#) module.

## Related Information

[Available Plans in the Kyma Environment \[page 2998\]](#)

[Provisioning and Updating Parameters in the Kyma Environment \[page 3002\]](#)

[Configure a Custom Identity Provider for Kyma \[page 3134\]](#)

[Assign Roles in the Kyma Environment \[page 3137\]](#)

[Add and Delete a Kyma Module \[page 3012\]](#)

[Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)

[Creating SAP BTP, Kyma runtime via the SAP BTP cli](#)

### 6.4.2 Managing Kyma Runtime Using the Provisioning Service API

The SAP Cloud Management service (technical name: `cis`) provides the Provisioning Service API to create and manage available environments. Use the Provisioning Service API to automatically manage and access SAP BTP, Kyma runtime.

## Prerequisites

- Your subaccount must have entitlements for SAP BTP, Kyma runtime and the SAP Cloud Management service for SAP BTP. See [Managing Entitlements and Quotas Using the Cockpit \[page 2185\]](#).
- Command line interface (CLI) tools:
  - [kubectl](#)
  - [jq](#)
  - [curl](#)
- SAP BTP command line interface (btp CLI). See [Download and Start Using the btp CLI Client \[page 2323\]](#).

## Context

To manage a Kyma instance automatically, create a Kyma service binding. The Kyma service binding enables getting a Kyma kubeconfig, which in turn allows for accessing a Kyma cluster, deploying applications, running tests, and deleting the resources in a fully automated way.

## Procedure

1. Provision the SAP Cloud Management service instance with the local plan and create a binding to get the credentials for the Provisioning Service API. To do that, you can use:

- SAP BTP cockpit, as described in [Getting an Access Token for SAP Cloud Management Service APIs \[page 2380\]](#).
- btp CLI and follow these steps:

- a. Set the `CIS_INSTANCE_NAME` environment variable with the name of the SAP Cloud Management service instance.

```
export CIS_INSTANCE_NAME={CIS_INSTANCE_NAME}
```

- b. Provision the SAP Cloud Management service instance with the Client Credentials grant type passed as parameters.

```
btp create services/instance --offering-name cis --plan-name local --name ${CIS_INSTANCE_NAME} --parameters {\"grantType\":\"clientCredentials\"}
```

- c. Create a binding for the instance.

```
btp create services/binding --name ${CIS_INSTANCE_NAME}-binding --instance-name ${CIS_INSTANCE_NAME}
```

2. Set the `CLIENT_ID`, `CLIENT_SECRET`, `UAA_URL`, and `PROVISIONING_SERVICE_URL` environment variables using the credentials from the binding stored in the `clientid`, `clientsecret`, `url`, and `provisioning_service_url` fields. Use the btp CLI to get the credentials.

```
export CLIENT_ID=$(btp --format json get services/binding --name ${CIS_INSTANCE_NAME}-binding | jq -r '.credentials.uaa.clientid')
export CLIENT_SECRET=$(btp --format json get services/binding --name ${CIS_INSTANCE_NAME}-binding | jq -r '.credentials.uaa.clientsecret')
export UAA_URL=$(btp --format json get services/binding --name ${CIS_INSTANCE_NAME}-binding | jq -r '.credentials.uaa.url')
export PROVISIONING_SERVICE_URL=$(btp --format json get services/binding --name ${CIS_INSTANCE_NAME}-binding | jq -r '.credentials.endpoints.provisioning_service_url')
```

3. Get the access token for the Provisioning Service API using the client credentials.

```
TOKEN=$(curl -s -X POST "${UAA_URL}/oauth/token" -H "Content-Type: application/x-www-form-urlencoded" -u "${CLIENT_ID}:${CLIENT_SECRET}" --data-urlencode "grant_type=client_credentials" | jq -r '.access_token')
```

4. Check if Kyma runtime is available for provisioning.

```
curl -s "$PROVISIONING_SERVICE_URL/provisioning/v1/availableEnvironments" -H "accept: application/json" -H "Authorization: bearer $TOKEN" | jq
```

- Set the `ENVIRONMENT_TYPE` and `SERVICE_NAME` environment variables to const values kyma and kymaruntime, and provide values for the `NAME`, `REGION`, `PLAN`, and `USER_ID` environment variables.

```
export ENVIRONMENT_TYPE="kyma"
export SERVICE_NAME="kymaruntime"
export NAME={RUNTIME_NAME}
export REGION={CLUSTER_REGION}
export PLAN={KYMA_RUNTIME_PLAN_NAME}
export USER_ID={USER_ID}
```

- Provision the Kyma runtime and save the instance ID in the `INSTANCE_ID` environment variable.

```
INSTANCE_ID=$(curl -s -X POST "$PROVISIONING_SERVICE_URL/provisioning/v1/environments" -H "accept: application/json" -H "Authorization: bearer $TOKEN" -H "Content-Type: application/json" -d "{\"environmentType\": \"$ENVIRONMENT_TYPE\", \"parameters\": {\"name\": \"$NAME\", \"region\": \"$REGION\"}, \"planName\": \"$PLAN\", \"serviceName\": \"$SERVICE_NAME\", \"user\": \"$USER_ID\"} | jq -r '.id'")
```

- Optional:** Set the `EXPIRATION_SECONDS` environment variable to the number of seconds (from 600 to 7200) after which a binding expires.

```
export EXPIRATION_SECONDS={EXPIRATION_SECONDS}
```

- After the provisioning is completed, create the binding and save the binding ID in the `BINDING_ID` environment variable.

```
[ -z "$EXPIRATION_SECONDS" ] && \
BINDING_ID=$(curl -sS -D - -X PUT "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID/bindings" -H "accept: application/json" -H "Authorization: bearer $TOKEN" -H "Content-Type: application/json" -d "{\"parameters\": {\"expiration_seconds\": 600}}" -o /dev/null | sed -n 's/^.*location: //p' | sed 's/\r$/\g') || \
BINDING_ID=$(curl -sS -D - -X PUT "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID/bindings" -H "accept: application/json" -H "Authorization: bearer $TOKEN" -H "Content-Type: application/json" -d "{\"parameters\": {\"expiration_seconds\": $EXPIRATION_SECONDS}}" -o /dev/null | sed -n 's/^.*location: //p' | sed 's/\r$/\g')
```

## ⓘ Note

You can have a maximum of 10 non-expired bindings. If you try to create more, you get the message stating that you've reached the maximum number of non-expired bindings.

- Get the binding credentials and save them in a kubeconfig file.

```
curl -s -X GET "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID/bindings/$BINDING_ID" -H "accept: application/json" -H "Authorization: bearer $TOKEN" | jq -r '.credentials.kubeconfig' > kubeconfig.yaml
```

- To access the cluster through kubectl, set the `KUBECONFIG` environment variable to the path of the kubeconfig file.

```
export KUBECONFIG=kubeconfig.yaml
```

- Verify the connection to the cluster. Run a kubectl command to get Pods:

```
kubectl get pods
```

kubectl should return the list of Pods in the default namespace running in the cluster, which means that the cluster is accessible.

12. **Optional:** To view the details of the binding you have created, list all bindings for the instance.

```
curl -s "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID/bindings" -H "accept: application/json" -H "Authorization: bearer $TOKEN"
```

13. **Optional:** For extra security, revoke the credentials by deleting the binding sooner than it is set to expire in the EXPIRATION\_SECONDS environment variable.

```
curl -s -X DELETE "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID/bindings/$BINDING_ID" -H "accept: application/json" -H "Authorization: bearer $TOKEN"
```

Try to access the cluster using kubectl. The connection should be refused, which means that the binding was successfully deleted and credentials revoked.

```
kubectl get pods
```

#### ⓘ Note

If you skip this step, the binding is automatically deleted after the maximum allowed expiration time (7200 seconds) passes.

## Next Steps

To deprovision Kyma runtime, run:

```
curl -s -X DELETE "$PROVISIONING_SERVICE_URL/provisioning/v1/environments/$INSTANCE_ID" -H "accept: application/json" -H "Authorization: bearer $TOKEN"
```

#### ⓘ Note

You can delete the runtime independently of the bindings. Existing bindings do not block the runtime deprovisioning.

## Related Information

[Account Administration Using APIs of the SAP Cloud Management Service \[page 2378\]](#)

## 6.4.3 Available Plans in the Kyma Environment

Depending on your global account type, you have access to a different plan that specifies the cluster parameters for the Kyma environment.

### Trial

The technical name of the plan is `trial`. For details on the trial cluster specification, see [Scope and Limitations \[page 227\]](#).

#### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Trial Plan Specification

Parameter	Description	Supported Operation	More Information
<i>Cluster Name*</i> btp CLI parameter: <code>name</code>	Defines the name of your cluster.	Provisioning	<a href="#">Cluster Name* [page 3004]</a>
<i>Modules</i> btp CLI parameter: <code>modules</code>	Defines which Kyma modules are provisioned in your cluster.	Provisioning	<a href="#">Modules [page 3006]</a>
<i>OpenID Connect</i> btp CLI parameter: <code>oidc</code>	Provides a custom OpenID Connect (OIDC) configuration.	Provisioning Updating	<a href="#">OpenID Connect (OIDC) [page 3009]</a>
<i>Administrators</i> btp CLI parameter: <code>administrators</code>	Specifies the list of runtime administrators.	Provisioning Updating	<a href="#">Administrators [page 3003]</a>

### Free

The technical name of the plan is `free`. Using the free service plans for Kyma allows you to try out services in global accounts without any additional cost for 30 days. However, the free model account has certain limitations.

- The free plan offers you a one-node cluster and is only available on AWS.
- Only community support is available for free tier service plans and these are not subject to SLAs.
- Before you enable the Kyma environment, you must first assign it as an entitlement to your subaccount.
- You can use the free plan only once in a global account for up to 30 days.

### → Remember

Once you have started the free plan in a subaccount, you cannot use it in another subaccount within the same global account even though the 30-day period has not ended.

- The services you plan to use must be available in the same region as the subaccount for the Kyma runtime. If necessary, change the default subaccount region.
- The upgrade to the paid plan is not yet supported.

For more information, read [Using Free Service Plans \[page 91\]](#).

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Free Plan Specification

Parameter	Description	Supported Operation	More Information
<i>Cluster Name*</i> btp CLI parameter: name	Defines the name of your cluster.	Provisioning	<a href="#">Cluster Name* [page 3004]</a>
<i>Region*</i> btp CLI parameter: region	Defines a region (set of data-centers) where your cluster runs.	Provisioning	<a href="#">Region* [page 3010]</a>
<i>Modules</i> btp CLI parameter: modules	Defines which Kyma modules are provisioned in your cluster.	Provisioning	<a href="#">Modules [page 3006]</a>
<i>Networking</i> btp CLI parameter: networking	Provides a custom IP range for worker nodes.	Provisioning	<a href="#">Networking [page 3008]</a>
<i>OpenID Connect</i> btp CLI parameter: oidc	Provides a custom Open ID Connect (OIDC) configuration.	Provisioning Updating	<a href="#">OpenID Connect (OIDC) [page 3009]</a>
<i>Administrators</i> btp CLI parameter: administrators	Specifies the list of runtime administrators.	Provisioning Updating	<a href="#">Administrators [page 3003]</a>

## Standard: Amazon Web Services, Google Cloud, and Microsoft Azure

The technical names of the standard enterprise plans are `aws`, `gcp`, and `azure`. They offer highly available Kubernetes clusters, where the Kubernetes and Kyma configurations are optimized for production use cases. The Kubernetes worker nodes are deployed in three availability zones of the respective [cloud region \[page 41\]](#), and thus can provide zone level failure tolerance for Kyma and applications deployed on Kyma runtime.

The [Kubernetes control plane](#) is also hosted in three availability zones of the respective region.

While the high availability is guaranteed for Kubernetes and native Kyma components, by default it's not guaranteed for the customer's own applications. To guarantee high availability for your applications deployed on Kyma, you must manually configure these applications.

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Standard Plan Specification

Parameter	Description	Supported Operation	More Information
<i>Cluster Name*</i> btp CLI parameter: <code>name</code>	Defines the name of your cluster.	Provisioning	<a href="#">Cluster Name* [page 3004]</a>
<i>Region*</i> btp CLI parameter: <code>region</code>	Defines a region (set of data-centers) where your cluster runs.	Provisioning	<a href="#">Region* [page 3010]</a>
<i>Machine Type</i> btp CLI parameter: <code>machineType</code>	Specifies the provider-specific virtual machine type.	Provisioning Updating	<a href="#">Machine Type [page 3005]</a>
<i>Auto Scaler Min</i> btp CLI parameter: <code>autoScalerMin</code>	Specifies the minimum number of virtual machines to create.	Provisioning Updating	<a href="#">Auto Scaler Min [page 3004]</a>
<i>Auto Scaler Max</i> btp CLI parameter: <code>autoScalerMax</code>	Specifies the maximum number of virtual machines to create.	Provisioning Updating	<a href="#">Auto Scaler Max [page 3003]</a>
<i>Modules</i> btp CLI parameter: <code>modules</code>	Defines which Kyma modules are provisioned in your cluster.	Provisioning	<a href="#">Modules [page 3006]</a>
<i>Networking</i> btp CLI parameter: <code>networking</code>	Provides a custom IP range for worker nodes.	Provisioning	<a href="#">Networking [page 3008]</a>
<i>OpenID Connect</i> btp CLI parameter: <code>oidc</code>	Provides a custom Open ID Connect (OIDC) configuration.	Provisioning Updating	<a href="#">OpenID Connect (OIDC) [page 3009]</a>
<i>Administrators</i> btp CLI parameter: <code>administrators</code>	Specifies the list of runtime administrators.	Provisioning Updating	<a href="#">Administrators [page 3003]</a>

## Kyma Test Demo and Development (Azure Lite)

The technical name of the plan is `azure_lite`. The Kyma Test Demo and Development plan is offered to Partners, within the [Pay-As-You-Go for SAP BTP for cloud test, demo, and development](#) commercial model. You can use this plan for testing, development, and demo purposes. In the SAP BTP cockpit, the plan is called "Kyma Runtime Partner TDD".

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Kyma Test Demo and Development Plan Specification

Parameter	Description	Supported Operation	More Information
<a href="#">Cluster Name*</a> btp CLI parameter: <code>name</code>	Defines the name of your cluster.	Provisioning	<a href="#">Cluster Name* [page 3004]</a>
<a href="#">Region*</a> btp CLI parameter: <code>region</code>	Defines a region (set of data-centers) where your cluster runs.	Provisioning	<a href="#">Region* [page 3010]</a>
<a href="#">Machine Type</a> btp CLI parameter: <code>machineType</code>	Specifies the provider-specific virtual machine type.	Provisioning Updating	<a href="#">Machine Type [page 3005]</a>
<a href="#">Auto Scaler Min</a> btp CLI parameter: <code>autoScalerMin</code>	Specifies the minimum number of virtual machines to create.	Provisioning Updating	<a href="#">Auto Scaler Min [page 3004]</a>
<a href="#">Auto Scaler Max</a> btp CLI parameter: <code>autoScalerMax</code>	Specifies the maximum number of virtual machines to create.	Provisioning Updating	<a href="#">Auto Scaler Max [page 3003]</a>
<a href="#">Modules</a> btp CLI parameter: <code>modules</code>	Defines which Kyma modules are provisioned in your cluster.	Provisioning	<a href="#">Modules [page 3006]</a>
<a href="#">Networking</a> btp CLI parameter: <code>networking</code>	Provides a custom IP range for worker nodes.	Provisioning	<a href="#">Networking [page 3008]</a>
<a href="#">OpenID Connect</a> btp CLI parameter: <code>oidc</code>	Provides a custom Open ID Connect (OIDC) configuration.	Provisioning Updating	<a href="#">OpenID Connect (OIDC) [page 3009]</a>

Parameter	Description	Supported Operation	More Information
<code>Administrators</code> btp CLI parameter: <code>administrators</code>	Specifies the list of runtime administrators.	Provisioning Updating	<a href="#">Administrators [page 3003]</a>

## Related Information

[Provisioning and Updating Parameters in the Kyma Environment \[page 3002\]](#)

[Regions for the Kyma Environment \[page 41\]](#)

[Kyma Modules \[page 74\]](#)

[Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)

[Service Plans and Metering for Kyma Runtime \[page 78\]](#)

[What Is the Consumption-Based Commercial Model? \[page 87\]](#)

## 6.4.4 Provisioning and Updating Parameters in the Kyma Environment

You can configure the cluster parameters in the Kyma environment.

### Overview

To configure the cluster parameters, you can use your preferred interface, the SAP BTP cockpit or the SAP BTP command line interface.

#### → Remember

The parameters marked with an asterisk "\*" are mandatory.

These are the configurable cluster parameters:

- [Administrators \[page 3003\]](#)
- [Auto Scaler Max \[page 3003\]](#)
- [Auto Scaler Min \[page 3004\]](#)
- [Cluster Name\\* \[page 3004\]](#)
- [Machine Type \[page 3005\]](#)
- [Modules \[page 3006\]](#)
- [Networking \[page 3008\]](#)
- [OpenID Connect \(OIDC\) \[page 3009\]](#)

- Region\* [page 3010]

To see which parameters are available for configuration in a particular plan, go to [Available Plans in the Kyma Environment](#) [page 2998].

## Administrators

The *Administrators* (`administrators`) parameter is an array of strings. It is a provisioning and updating parameter, which specifies the list of runtime administrators. Complete the list with the administrators' email addresses as shown in this example:

```
"administrators": [
    "example_1@mail.com",
    "example_2@mail.com",
    "example_3@mail.com"
]
```

## Auto Scaler Max

The *Auto Scaler Max* (`autoScalerMax`) is an integer parameter, which specifies the maximum number of virtual machines you can create.

Auto Scaler Max Parameter

Plan	Supported Operation	Default Value	Maximum Value	Allowed Input
Standard: Amazon Web Services (technical name: <code>aws</code> )	Provisioning Updating	20	300	Number between 3 and 300, but greater than or equal to <a href="#">Auto Scaler Min</a> .
Standard: Google Cloud (technical name: <code>gcp</code> )				
Standard: Microsoft Azure (technical name: <code>azure</code> )				
Kyma Test Demo and Development (Azure Lite) (technical name: <code>azure_lite</code> )	Provisioning Updating	10	40	Number between 2 and 40, but greater than or equal to <a href="#">Auto Scaler Min</a> .

The default JSON input is:

```
"autoScalerMax": 20
```

## Auto Scaler Min

The [Auto Scaler Min](#) (`autoScalerMin`) is an integer parameter, which specifies the minimum number of virtual machines you can create.

Auto Scaler Min Parameter

Plan	Supported Operation	Default Value	Minimum Value	Allowed Input
Standard: Amazon Web Services (technical name: <code>aws</code> )	Provisioning Updating	3	3	Number between 3 and the current value set in <a href="#">Auto Scaler Max</a> .
Standard: Google Cloud (technical name: <code>gcp</code> )				
Standard: Microsoft Azure (technical name: <code>azure</code> )				
Kyma Test Demo and Development (Azure Lite) (technical name: <code>azure_lite</code> )	Provisioning Updating	2	2	Number between 2 and the current value set in <a href="#">Auto Scaler Max</a> .

Here is an example of the JSON input:

```
"autoScalerMin": 3
```

## Cluster Name\*

The [Cluster Name](#) (`name`) is a string, which provides the name of your cluster.

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

Cluster Name Parameter

Parameter	Supported Operation	Default Value	Allowed Input
<a href="#">Cluster Name*</a> btp CLI parameter: <code>name</code> type: string	Provisioning	Automatically generated as your Subaccount's Subdomain.	Short string (up to 32 characters) that contains only alphanumeric characters (A-Z, a-z, 0-9), periods, underscores, and hyphens. It can't contain white spaces.

## Machine Type

The *Machine Type* (`machineType`) parameter is a string, which specifies the provider-specific virtual machine type.

Machine Type Parameter

Plan	Supported Operation	Default Value	Allowed Input	Virtual Machine Size
Standard: Amazon Web Services technical name: aws	Provisioning Updating	<code>m6i.large</code>	<code>m6i.large</code>	2vCPU, 8GB RAM
			<code>m6i.xlarge</code>	4vCPU, 16GB RAM
			<code>m6i.2xlarge</code>	8vCPU, 32GB RAM
			<code>m6i.4xlarge</code>	16vCPU, 64GB RAM
			<code>m6i.8xlarge</code>	32vCPU, 128GB RAM
			<code>m6i.12xlarge</code>	48vCPU, 192GB RAM
			<code>m5.large</code>	2vCPU, 8GB RAM
			<code>m5.xlarge</code>	4vCPU, 16GB RAM
			<code>m5.2xlarge</code>	8vCPU, 32GB RAM
			<code>m5.4xlarge</code>	16vCPU, 64GB RAM
Standard: Google Cloud technical name: gcp	Provisioning Updating	<code>n2-standard-2</code>	<code>n2-standard-2</code>	2vCPU, 8GB RAM
			<code>n2-standard-4</code>	4vCPU, 16GB RAM
			<code>n2-standard-8</code>	8vCPU, 32GB RAM
			<code>n2-standard-16</code>	16vCPU, 64GB RAM
			<code>n2-standard-32</code>	32vCPU, 128GB RAM
			<code>n2-standard-48</code>	48vCPU, 192B RAM
			<code>Standard_D2s_v5</code>	2vCPU, 8GB RAM
			<code>Standard_D4s_v5</code>	4vCPU, 16GB RAM
			<code>Standard_D8s_v5</code>	8vCPU, 32GB RAM
			<code>Standard_D16s_v5</code>	16vCPU, 64GB RAM
Standard: Microsoft Azure technical name: azure	Provisioning Updating	<code>Standard_D2s_v5</code>	<code>Standard_D32s_v5</code>	32vCPU, 128GB RAM
			<code>Standard_D48s_v5</code>	48vCPU, 192GB RAM
			<code>Standard_D64s_v5</code>	64vCPU, 256GB RAM
			<code>Standard_D4_v3</code>	4vCPU, 16GB RAM

Plan	Supported Operation	Default Value	Allowed Input	Virtual Machine Size
			Standard_D8_v3	8vCPU, 32GB RAM
			Standard_D16_v3	16vCPU, 64GB RAM
			Standard_D32_v3	32vCPU, 128GB RAM
			Standard_D48_v3	48vCPU, 192GB RAM
			Standard_D64_v3	64vCPU, 256GB RAM
Kyma Test Demo and Development (Azure Lite)	Provisioning	Standard_D4s_v5	Standard_D4s_v5	4vCPU, 16GB RAM
	Updating		Standard_D4_v3	4vCPU, 16GB RAM
technical name: azure_lite				

Here is an example of the input for the *Machine Type* parameter:

```
"machineType": "m5.xlarge"
```

## Modules

With the *Modules* (`modules`) object, you can define which Kyma modules you want to provision in your cluster. You can also use it to create a cluster without any modules.

### ⓘ Note

API for module configuration is built on the `oneOf` feature from the JSON schema. If the `modules` object is passed to API, it must have only one valid option: *Default* (default) or *Custom* (list). Even if you remove the `modules` object from the JSON schema, the default Kyma modules are provisioned in your cluster.

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Modules Parameters

Nested Parameter	Description	Supported Operation	Default Value
<i>Default</i> btp CLI parameter: <code>default</code> type: boolean	Defines whether to use the default list of Kyma modules. Check the <a href="#">Default Kyma Modules</a> to find out which modules are in the default modules list.	Provisioning	true

Nested Parameter	Description	Supported Operation	Default Value
<i>Custom</i>	Defines a custom list of Kyma modules.	Provisioning	[ ] (empty array)
btp CLI parameter: list type: array of objects	Leave your custom list of Kyma modules empty if you don't want any modules provisioned.		

The *Custom* (list) parameter includes three nested parameters: *Name\**, *Channel*, and *Custom Resource Policy*.

#### Custom list Parameters

Nested Parameter	Description	Supported Operation	Default Value
<i>Name*</i>	Look up the available Kyma modules and their technical names at <a href="#">Kyma Modules</a> [page 74].	Provisioning	n/a
btp CLI parameter: name type: string			
<i>Channel</i>	Defines the preferred release channel. The default is <code>regular</code> . The other option is <code>fast</code> .	Provisioning	"" (an empty string)
btp CLI parameter: <code>channel</code> type: string			
<i>Custom Resource Policy</i>	Defines how a module's custom resource configuration is handled during enablement and reconciliation.	Provisioning	"" (an empty string)
btp CLI parameter: <code>customResourcePolicy</code> Y type: string	By default, it is set to <code>CreateAndDelete</code> and allows the creation or deletion of a module's resource.  If you set it to <code>Ignore</code> , a module's resource is not created.		

You have the default Kyma modules provisioned in your cluster if you do not provide the `modules` object in the JSON payload, or if you use the following input:

```
"modules": {
    "default": true
}
```

Here is an example of JSON input for a custom list of Kyma modules:

```
"modules": {
    "list": [
        {
            "name": "btp-operator"
        },
        ...
    ]
}
```

```

        {
            "name": "keda",
            "customResourcePolicy": "CreateAndDelete",
            "channel": "fast"
        }
    ]
}

```

Applying the following values results in Kyma runtime provisioning without any Kyma modules.

```
"modules": {
    "list": []
}
```

```
"modules": {
    "default": false
}
```

## Networking

The [Networking](#) (networking) object provides networking configuration. These values are immutable and cannot be updated later.

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### Networking Parameters

Nested Parameter	Description	Supported Operation	Default Value	Allowed Input
<i>CIDR range for Nodes*</i> btp CLI parameter: <code>nodes</code> type: string	Defines a custom IP range for worker Nodes.	Provisioning	10.250.0.0/22	CIDR range for Nodes must not overlap with the following CIDRs: 10.96.0.0/13, 10.104.0.0/13, 10.243.128.0/17, 10.242.0.0/16, 10.243.0.0/17, 10.64.0.0/11, 10.254.0.0/16, 10.243.0.0/16
<i>CIDR range for Pods</i> btp CLI parameter: <code>pods</code> type: string	Defines a custom IP range for Pods.	Provisioning	10.96.0.0/13	CIDR range for Pods must not overlap with the following CIDRs: 10.96.0.0/13, 10.104.0.0/13, 10.243.128.0/17, 10.242.0.0/16, 10.243.0.0/17, 10.64.0.0/11, 10.254.0.0/16, 10.243.0.0/16

Nested Parameter	Description	Supported Operation	Default Value	Allowed Input
<i>CIDR range for Services</i>  btp CLI parameter: services  type: string	Defines a custom IP range for Services.	Provisioning	10.104.0.0/13	CIDR range for Services must not overlap with the following CIDRs: 10.96.0.0/13, 10.104.0.0/13, 10.243.128.0/17, 10.242.0.0/16, 10.243.0.0/17, 10.64.0.0/11, 10.254.0.0/16, 10.243.0.0/16

Here is the default JSON input for the [Networking](#) object:

```
"networking": {
    "nodes": "10.250.0.0/22"
    "pods": "10.96.0.0/13"
    "services": "10.104.0.0/13"
}
```

## OpenID Connect (OIDC)

The [OpenID Connect](#) (oidc) object allows you to provide OIDC configuration. If you do not provide the oidc object or any custom values in the provisioning request, the default OIDC configuration is used. If you do not provide the oidc object in the update request or leave all object's properties empty, the saved OIDC configuration remains unchanged.

### → Remember

The parameters marked with an asterisk "\*" are mandatory.

#### OIDC Parameters

Nested Parameter	Description	Supported Operation	Default Value	Allowed Input
<i>Client ID*</i>  btp CLI parameter: clientID  type: string	The client ID for the OpenID Connect client.	Provisioning  Updating	n/a	n/a
<i>Issuer URL*</i>  btp CLI parameter: issuerURL  type: string	Provides the URL of the OpenID issuer.	Provisioning  Updating	n/a	n/a

Nested Parameter	Description	Supported Operation	Default Value	Allowed Input
<i>Groups Claim</i> btp CLI parameter: <code>groupsClaim</code>	If provided, specifies the name of a custom OIDC claim for specifying user groups.  type: string	Provisioning Updating	groups	n/a
<i>Username Claim</i> btp CLI parameter: <code>usernameClaim</code>	Provides the OpenID claim to use as the user name.  type: string	Provisioning Updating	n/a	Only HTTPS scheme is accepted.
<i>Signing Algs</i> btp CLI parameter: <code>signingAlgs</code>	Provides the OIDC signing algorithms for Kyma runtime.  type: array	Provisioning Updating	RS256	Comma separated list of allowed JOSE asymmetric signing algorithms, for example, RS256, ES256.
<i>Username Prefix</i> btp CLI parameter: <code>usernamePrefix</code>	Provides an OIDC user-name prefix for Kyma runtime.  type: string	Provisioning Updating	If not provided, user-name claims other than an email address are prefixed by the issuer URL to avoid clashes.	To skip any prefixing, provide the value "-" (dash character without additional characters).

The following example shows the default configuration of the [OIDC](#) parameter. When you want to revert your changes to the default settings, copy and paste the following values:

```
"oidc": {
  "clientID": "12b13a26-d993-4d0c-aa08-5f5852bbdff6",
  "groupsClaim": "groups",
  "issuerURL": "https://kyma.accounts.ondemand.com",
  "signingAlgs": ["RS256"],
  "usernameClaim": "sub",
  "usernamePrefix": "-"
}
```

## Region\*

[Region\\*](#) (`region`) is a mandatory string parameter, which defines a region where your cluster runs.

### Region Parameter

Plan	Supported Operation	Region Technical Key	Region Name
Standard: Amazon Web Services	Provisioning	eu-central-1	Europe (Frankfurt)
		eu-west-2	Europe (London)
technical name: aws		ca-central-1	Canada (Montreal)

Plan	Supported Operation	Region Technical Key	Region Name
Free		sa-east-1	Brazil (São Paulo)
technical name: free		us-east-1	US East (VA)
		us-west-1	US West (N. California)
		ap-northeast-1	Japan (Tokyo)
		ap-northeast-2	South Korea (Seoul)
		ap-south-1	India (Mumbai)
		ap-southeast-1	Singapore
		ap-southeast-2	Australia (Sydney)
Standard: Google Cloud	Provisioning	europe-west3	Europe (Frankfurt)
technical name: gcp		asia-south1	India (Mumbai)
		us-central1	US Central (IA)
		asia-northeast2	Japan (Osaka)
		me-central2	KSA (Dammam)
		me-west1	Israel (Tel Aviv)
		australia-southeast1	Australia (Sydney)
		southamerica-east1	Brazil (São Paulo)
Standard: Microsoft Azure	Provisioning	eastus	US East (VA)
technical name: azure		centralus	US Central (IA)
Kyma Test Demo and Development (Azure Lite)		westus2	US West (WA)
technical name: azure_lite		uksouth	UK South (London)
		northeurope	North EU (Ireland)
		westeurope	Europe (Netherlands)
		japaneast	Japan (Tokyo)
		southeastasia	Singapore
		australiaeast	Australia (Sydney)
		switzerlandnorth	Switzerland (Zurich)
		brazilsouth	Brazil (São Paulo)

Here is an example of the JSON input for the [Region](#) parameter:

```
"region": "us-east-1"
```

## Related Information

[Available Plans in the Kyma Environment \[page 2998\]](#)

[Regions for the Kyma Environment \[page 41\]](#)

[Kyma Modules \[page 74\]](#)

[Account Administration Using the SAP BTP Command Line Interface \(btp CLI\) \[page 2321\]](#)

## 6.4.5 Add and Delete a Kyma Module

To use a Kyma module, you must add it first. Use Kyma dashboard or kubectl to do that. If you don't need the module anymore, delete it to save resources.

### 6.4.5.1 Add and Delete a Kyma Module Using Kyma Dashboard

Use Kyma dashboard to add and delete a Kyma module.

#### Context

Follow this procedure to easily add a module from the dashboard's *Cluster Details* view.

#### Procedure

1. Log in to Kyma dashboard. The URL is in the *Overview* section of your subaccount.
2. Choose *Modify Modules*, and select *Add*.
3. In the *Add Modules* section, check the modules you want to add, and select *Add*.
4. **Optional:** At the module level, you can overwrite the default release channel for the modules you are adding: Under the *Advanced* options, choose your preferred release channel.

#### Results

This process may take a while, depending on the number of modules. The operation was successful when the module status changes to **READY**.

## Next Steps

- To configure your module, use the module CR that you can find in the module repository.
- To delete a module, choose [Modify Modules](#), and click on the trash icon next to the module you want to delete.

### 6.4.5.2 Add and Delete a Kyma Module Using kubectl

Use kubectl to add or delete a Kyma module in the default channel.

#### Context

To add a module using [kubectl](#) , perform the following steps:

#### Procedure

1. Run the following command:

```
kubectl edit kyma default -n kyma-system
```

2. In the editor, add a module on your cluster in `spec.modules` by adding the module's name:

```
spec:  
  modules:  
    - name: {NAME_OF_THE_MODULE}
```

3. Save the changes.

You should see the following message:

```
kyma.operator.kyma-project.io/default edited
```

## Next Steps

- To configure your module, edit the module CR.
- To delete a module, remove its name from the editor.

## Related Information

[Kyma's Modular Approach \[page 72\]](#)

[Kyma Release Channels \[page 72\]](#)

[Kyma Modules \[page 74\]](#)

## 6.4.6 Access Kyma Application Logs

Get insights into your applications, microservices, and Functions by viewing the respective logs. To check out real-time logs immediately, use the Kubernetes functionalities - either in Kyma dashboard, or with kubectl.

### Context

- If you prefer to see the logs in your terminal, replace the placeholders in the following command and run it:

```
kubectl logs <{POD_NAME}> --namespace <{NAMESPACE_NAME}> --container  
<{CONTAINER_NAME}>
```

- To see real-time logs in Kyma dashboard, follow these steps:

### Procedure

1. Open Kyma dashboard and select the namespace.
2. Access the Pod.
3. Find the container and select *View Logs*.

### Results

You have access to short-term logs of every container in your Kyma runtime.

### Next Steps

#### → Recommendation

If you want long-term storage of logs with advanced querying, dashboarding, and alerting capabilities, set up SAP Cloud Logging to ingest observability data with Kyma runtime.

## Related Information

[Auditing and Logging Information in Kyma \[page 3139\]](#)

[Create an SAP Cloud Logging Instance through SAP BTP Service Operator](#)

[SAP Cloud Logging: Ingest via Kyma Runtime](#)

## 6.4.7 Kyma Environment Backup

The user load in a Kyma cluster typically consists of various Kubernetes objects and volumes.

- Objects hold the configuration of user's application applied to the API server using Kubernetes resources
- Volumes hold the actual data that a user may store in a persistent way

Kyma does not provide backup, neither for objects nor volumes, that can replace the configuration or data from a specific point in time.

## Object Backup for Kubernetes Configuration

Kyma environment relies on the managed Kubernetes cluster for periodic backups of Kubernetes objects. It means that it is possible to automatically restore Kubernetes objects in case of inconsistencies or data corruption. However, Kyma environment does not provide backup that can replace the configuration from a specific point in time. Automatic backup doesn't include Kubernetes volumes.

For example, project "Gardener" uses etcd as the Kubernetes' backing store for all cluster data. This means that all Kubernetes objects are stored on etcd. Gardener uses periodic jobs to take major and minor snapshots of the etcd database. A major snapshot (including all the resources) takes place every day, and each minor snapshot (including only the changes in between) takes place every 5 minutes. If the etcd database experiences any problems, Gardener automatically restores the Kubernetes cluster using the latest snapshot.

### → Tip

Follow GitOps principles to reapply the desired state of your Kubernetes objects to any Kubernetes cluster at any time.

- GitOps means managing your Kubernetes objects using [Git](#).
- A Git repository becomes a single source of truth where you declare the desired state of your Kubernetes objects.
- Automation tooling of your choice pulls the desired state from that single source and applies the declared configuration to your cluster.

For more information, see [OpenGitOps](#).

## Volume Backup for Customer Data

Your customer data isn't backed up automatically. If you're using Kubernetes volumes to store data, we recommend that you use Kubernetes VolumeSnapshots to back up and recover your data. The backup and recovery using Kubernetes VolumeSnapshots are supported for AWS, Azure, and Google Cloud.

## Related Information

[Change Storage Size in Kyma \[page 3019\]](#)

## 6.4.8 Kyma Modules' Sizing

The following tables show the resource consumption, such as memory, CPU, and storage, of Kyma modules.

### API Gateway

API Gateway Module's Sizing

Infrastructure size	Memory	CPU
Small cluster minimum	212Mi	0.03
Large cluster minimum	416Mi	0.34

### Istio

Istio Module's Sizing

Infrastructure size	Memory	CPU
Small cluster minimum	352Mi	0.08
Large cluster minimum	1984Mi	0.61

### Application Connector

Application Connector Module's Sizing

Infrastructure size	Memory	CPU
Minimum	64-128Mi	0.2-0.5

## SAP BTP Operator

SAP BTP Operator Module's Sizing

Infrastructure size	Memory	CPU
Minimum	53Mi	0.02
Maximum	628Mi	1.2

## Keda

Keda Module's Sizing

Infrastructure size	Memory	CPU
Minimum	692Mi	0.62
Maximum	928Mi	1.8

## Serverless

Serverless Module's Sizing

Infrastructure size	Memory	CPU	Storage
Minimum	400Mi	0.03	
Maximum	2300Mi	1.7	Up to 20000Mi (for PVC for internal Docker registry)

## Telemetry

Resource consumption depends on the APIs (pipelines) you are using. For details, see [Module Lifecycle \[page 1925\]](#).

Using the Telemetry module without activating any pipelines results in the Telemetry Manager running with the following footprint:

Infrastructure size	Memory	CPU
Minimum	5Mi	0.01
Maximum	100Mi	0.1

Activation of the first `LogPipeline` causes the deployment of the log agent running an instance per node:

Infrastructure size	Memory	CPU
Minimum per node	50Mi	0.1
Maximum per node	1000Mi	1

Activation of the first `TracePipeline` causes the deployment of the trace gateway running with two replicas:

Infrastructure size	Memory	CPU
Minimum	2*32Mi	2*0.2
Maximum	2*2000Mi	2*1.2

Activation of the first `MetricPipeline` causes the deployment of the metric gateway running with two replicas and the metric agent running per node:

Infrastructure size	Memory	CPU
Minimum	2*30Mi + 50Mi per node	2*0.01 + 0.01 per node
Maximum	2*1000Mi + 1200Mi per node	2*1.00 + 1 per node

## NATS

By default, the NATS module creates a NATS-cluster with three nodes for data safety reasons. The values are given per NATS-cluster node:

NATS Module's Sizing

Infrastructure size	Memory	CPU	Storage
Minimum	64Mi	0.04	1000Mi
Maximum	1000Mi	0.5	1000Mi

## Eventing

Eventing Manager's Sizing

Infrastructure size	Memory	CPU
Minimum	128Mi	0.01
Maximum	512Mi	0.5

#### Eventing Publisher Proxy's Sizing

Infrastructure size	Memory	CPU	Max Publisher instances
Low EPS (up to 200 Events per second)	64-512Mi	0.04-0.5	2 (second instance for high availability)
High EPS (up to 3000 Events per second)	64-512Mi	0.04-0.5	3

## Related Information

[Sizing](#)

### 6.4.9 Change Storage Size in Kyma

If the amount of data for the applications in your Kyma environment grows, you can expand the storage size for your customer data by resizing the respective Persistent Volume Claim (PVC).

## Prerequisites

- Your user has the authorizations to change PVC resources.
- Your workload uses a Persistent Volume, managed by a PVC.

## Context

In a Kubernetes cluster, your customer data is stored in volumes. A “Persistent Volume Claim” (PVC) manages that volume for you and defines the size of storage. As the amount of customer data grows, you must adjust the PVC accordingly.

## Procedure

1. Go to Kyma dashboard and select the correct namespace and workload.
2. Find the PVC you want to resize and enter the desired new size of the volume.
3. Save your changes.

The PVC shows the condition type [Resizing](#). Wait until the PVC switches to condition type [FileSystemResizePending](#). This may take a few minutes.

4. Restart your workload by restarting all related Pods.

## Results

The storage size for your customer data has been adjusted to your needs.

## Next Steps

Remember to back up your customer data.

## Related Information

[Kubernetes: Expanding Persistent Volumes Claims](#) ↗

[Kyma Environment Backup \[page 3015\]](#)

# 7 Security

Use the security features and functions of SAP BTP to support the security policies of your organization.

## Security Recommendations

We provide a list with our recommendations for the configuration of our services. These recommendations help you to meet your compliance goals and secure your business.

See [SAP BTP Security Recommendations](#).

Our customer success organization uses these recommendations as a base to create a security baseline template.

For more information, go to <https://support.sap.com/sos> and choose  [Media Library](#)  [SAP CoE Security Services - Security Baseline Template](#).

## User Model

SAP BTP distinguishes between **platform users** (account management, custom development, and operations) and **business users** (for the applications).

See [User and Member Management \[page 104\]](#).

## Authorizations

You can configure authorizations using **roles** and **role collections** for your global account, subaccount, directory, or individual applications.

See [Security Administration: Managing Authentication and Authorization \[page 2202\]](#).

## Identity Providers

All users of SAP BTP are stored in identity providers, either in the default or in a custom identity provider. SAP BTP needs a copy of the user, sometimes called a shadow user. You assign the shadow user authorizations to access resources in SAP BTP. When a user authenticates, SAP BTP forwards the request to the identity provider.

For more information, see [Trust and Federation with Identity Providers \[page 2204\]](#).

### Note

For the China (Shanghai) region, a different default identity provider is used.

For more information, see this [blog article](#) on SAP Community.

## Default Identity Provider

We provide a default identity provider for both platform users and business users (in applications) at SAP BTP. The default identity provider enables single sign-on to your SAP applications and services.

Use the default identity provider as a preconfigured user store in your starter scenarios or for testing. You can also use the default identity provider as a backup identity provider if access to your custom identity provider fails.

See [Default Identity Provider \[page 2258\]](#).

## Identity Authentication Service

Identity Authentication service provides authentication and single sign-on in the cloud.

We recommend that you configure the Identity Authentication service as the identity provider and connect Identity Authentication to your own corporate identity provider. Identity Authentication provides features that the default identity provider doesn't, such as the ability to connect your corporate identity provider or to define security policies.

See [Trust and Federation with Identity Providers \[page 2204\]](#).

For more information about Identity Authentication, see [SAP Cloud Identity Services - Identity Authentication](#).

## Transport Layer Security (TLS) Connectivity Support

SAP BTP uses encrypted communication channels based on HTTPS/TLS, supporting TLS version 1.2 or higher.

### Note

TLS versions 1.0 and 1.1 are no longer supported.

Make sure you use HTTP clients (such as web browsers) that support TLS version 1.2 or higher for connecting to SAP BTP.

### Note

You can optionally use TLS 1.3 in the Custom Domain Manager. This option allows the use of TLS 1.3 with applications running on SAP BTP. It's not allowed to use TLS 1.3, for example for the SAP BTP cockpit or SAP Cloud Identity Services. These services are still using TLS 1.2.

See [What Is Custom Domain](#).

## Audit Logging

Use the Audit Log Retrieval API to view the audit logs stored for your subaccount. Use the audit log viewer to display the audit logs for your Cloud Foundry account, produced by SAP applications and services you've subscribed to. See [Audit Logging in the Cloud Foundry Environment \[page 2501\]](#).

## Credential Store

SAP Credential Store provides a repository for passwords and keys for applications that are running on SAP BTP, Cloud Foundry environment. It enables the applications to retrieve credentials and use them for authentication to external services, or to perform cryptographic operations and TLS communication.

See [SAP Credential Store](#).

## Malware Scanning

Use the SAP Malware Scanning service to scan business documents for malware. Integrate this service with your custom-developed apps running on the Cloud Foundry runtime. When your apps upload business documents, your apps can call the SAP Malware Scanning service to check for viruses or other malware.

For more information, see [SAP Malware Scanning Service](#).

## Related Information

[SAP Authorization and Trust Management Service \[page 3023\]](#)

[Audit Logging in the Cloud Foundry Environment \[page 2501\]](#)

[Principal Propagation \[page 3095\]](#)

[Data Protection and Privacy \[page 3109\]](#)

[Security in the Kyma Environment \[page 3120\]](#)

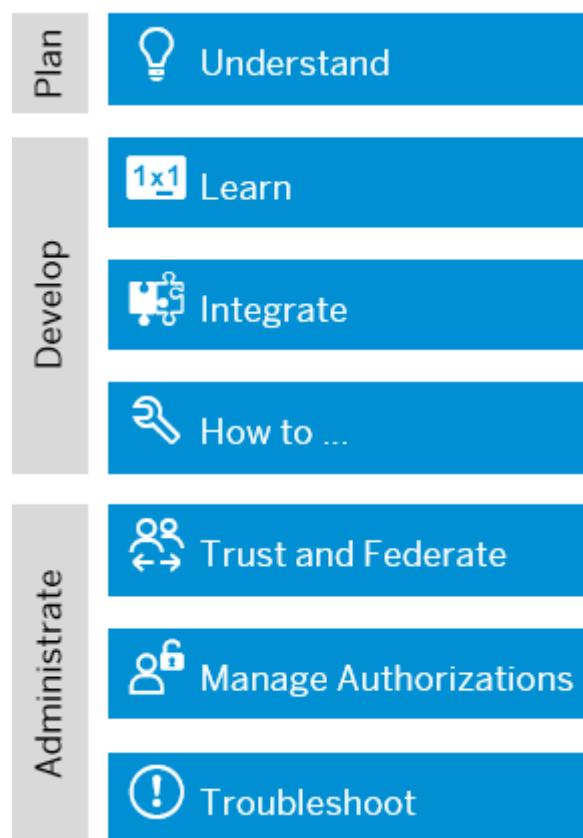
## 7.1 SAP Authorization and Trust Management Service

The global account and subaccounts get their users from identity providers. Administrators make sure that users can only access their dedicated subaccount by making sure that there is a dedicated trust relationship

only between the identity providers and the respective subaccounts. Developers configure and deploy application-based security artifacts containing authorizations, and administrators assign these authorizations using the SAP BTP cockpit.

### Note

Before you start, make yourself familiar with the sections about authentication and authorization of the SAP BTP Planning and Lifecycle-Management Guide. See the [Setting Up Your Security and Compliance Model](#) section.



- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im1 \[page 3025\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im2 \[page 3025\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im3 \[page 3026\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im4 \[page 3027\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im5 \[page 3028\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im6 \[page 3029\]](#)
- [#unique\\_368/unique\\_368\\_Connect\\_42\\_subsection-im7 \[page 3030\]](#)

Hold your pointer over a box for a description. Select a box to display more information.

## Overview of the SAP Authorization and Trust Management Service

Get a high-level overview of the concepts that underpin the SAP Authorization and Trust Management service for SAP BTP in the Cloud Foundry environment.

For more information, see [What Is the SAP Authorization and Trust Management Service? \[page 3030\]](#).

## Tutorials for the SAP Authorization and Trust Management Service

Follow the tutorials below to get familiar with the SAP Authorization and Trust Management service in the Cloud Foundry environment of SAP BTP.

### Tutorials for the SAP Authorization and Trust Management service in the Cloud Foundry environment

Language / Framework	Link
Learn how to secure a basic single-tenant Node.js application. Start with a Node.js application that uses the express framework and SAPUI5 to display a list of products and add the security components step by step.	<a href="#">SAP Developers</a>
Learn how to secure a basic java application. This tutorial starts with a Hello World Java application built with SAP Cloud SDK.	<a href="#">SAP Developers</a>
Learn how to secure microservices in SAP BTP using <code>spring-xsuaa</code> and Spring security. Furthermore, learn how to test the secured application using the <code>java-security-test</code> utilities.	<a href="#">GitHub</a>
Learn how to add multitenancy to a node.js application and make it available for other subaccounts using the SaaS Provisioning service and the XSUAA.	<a href="#">SAP Developers</a>
Learn how to secure microservices in SAP BTP. This sample provides J2EE Configuration using <code>web.xml</code> and uses the SAP Java Buildpack.	<a href="#">GitHub (SAP Java Buildpack version &lt;= 1.26.0)</a> <a href="#">GitHub (SAP Java Buildpack version &gt;=1.26.1)</a>

**Tutorials for the SAP Authorization  
and Trust Management service in the  
Cloud Foundry environment**

Language / Framework	Link
Learn how to build a cloud-native Node.js application that features secured service-to-service communication. The application shows you two different ways of securing service-to-service-communication (by propagating a business user or using a technical user).	<a href="#">GitHub</a>
Learn how to use the <code>java-security</code> library to perform JWT Validation as part of your Java application. Furthermore, learn how to test the secured application using the <code>java-security-test</code> utilities.	<a href="#">GitHub</a>
Learn in this reference application how the service fits into a complete architecture of microservices that interact with each other propagating user information.	<a href="#">GitHub</a>
Learn how to validate OAuth tokens using a Python library. Use this library to add authentication in your Python application.	<a href="#">GitHub</a>

**ⓘ Note**

This library isn't part of an SAP BTP license. However, it belongs to a related open source project.

## Principal Propagation

Exchange user ID information between systems or environments in SAP BTP.

### In This Section

- [Principal Propagation from the Cloud Foundry to the Neo Environment \[page 3102\]](#)
- [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 3095\]](#)

### Other Principal Propagation Scenarios

- [On-Premise User Store](#)
- [Principal Propagation to OAuth-Protected Applications](#)
- [Connectivity in the Cloud Foundry Environment: Principal Propagation](#)

- Connectivity in the Neo Environment: Principal Propagation

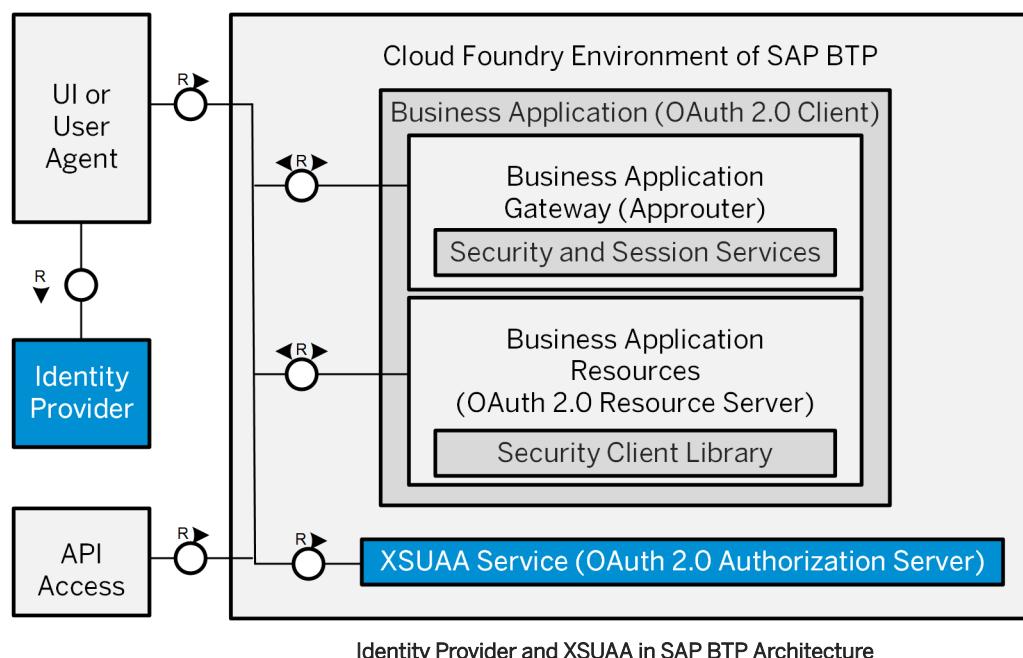
## Trust and Federation

SAP BTP supports identity federation, a concept of linking and reusing digital identities of a user base across loosely coupled systems. Identity federation frees applications on When setting up accounts you need to assign users. While we provide you with your first users from the default identity provider to get you started, your organization has identity providers that you want to integrate.SAP BTP as well as the platform itself from the need to obtain and store the credentials of users and to authenticate them. Instead, the user base is reused from identity providers, which support the administration of digital user identities, authentication, and authorizations in a centralized and decoupled manner. To enable communication between SAP BTP and identity providers, you must cross-configure the communication endpoints of the involved systems, establishing a trust relationship between them.

### → Recommendation

We recommend that you use a custom tenant of SAP Cloud Identity ServicesWhen setting up accounts you need to assign users. While we provide you with your first users from the default identity as identity provider and connect a potential corporate identity provider there. For platform users, the use of SAP Cloud Identity Services is mandatory. If you don't have a tenant yet, check [Getting a Tenant](#).

To connect your corporate identity provider to SAP Cloud Identity Services, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication and SAP Cloud Identity Services](#)



SAP Cloud Identity Services is a multitenancy-enabled identity provider for all SAP cloud applications and optionally on-premise applications. The service provides capabilities for authentication, single sign-on, authorizations, identity lifecycle management, and on-premise integration as well as self-services like self-registration or password reset.

SAP has its own SAP Cloud Identity Services tenant, SAP ID service. SAP ID service is the default identity provider of SAP BTP and where you register to get initial access to SAP BTP. Trust to SAP ID service is preconfigured by default.

We recommend that you request your own SAP Cloud Identity Services tenant (see [Getting a Tenant](#)). To establish trust with your identity provider, proceed as follows.

For business users:

- [Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2205\]](#)
- [Manually Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2224\]](#)
- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 2228\]](#)

For platform users:

- [Establish Trust and Federation of Custom Identity Providers for Platform Users \[page 2239\]](#)

For default identity provider:

- [Default Identity Provider](#)

#### ⓘ Note

How you assign users to their authorizations depends on the type of trust configuration. If you're using the default trust configuration via SAP ID service, you can assign users directly to role collections. For more information, see [Default Identity Provider \[page 2258\]](#).

However, if you're using a custom trust configuration as described in this topic, you can assign individual users or groups to role collections. Assigning users to their authorizations is part of application administration, which is described here. For more information, see [Mapping Role Collections in the Subaccount \[page 2281\]](#).

## Administration: Managing Authentication and Authorization

Application developers create and deploy application-based authorization artifacts for business users. Administrators use this model to manage roles, build role collections, and assign these collections to users or user groups. In this way, they control the users' permissions.

Setting Up Authorization Artifacts (Account Administrators)

Task	Links
Assign the role collection to the users provided by an identity provider	<a href="#">Working with Role Collections [page 2274]</a>
(If you do use a custom identity provider) Assign the role collections to user groups	<a href="#">Map Role Collections to User Groups [page 2281]</a>
Assign the role collections to users and user groups, manage attribute mappings	<a href="#">Mapping Role Collections in the Subaccount [page 2281]</a>
Create a role collection and assign roles to it	<a href="#">Define a Role Collection [page 2275]</a>

Task	Links
Use an existing role or create a new one using role templates	<a href="#">Add Roles to Role Collections on the Application Level [page 2293]</a>

## Troubleshooting

This section provides information on troubleshooting-related activities for the SAP Authorization and Trust Management service in the Cloud Foundry environment.

### → Tip

We also recommend that you regularly check the SAP Notes and Knowledge Base for component BC-CP-CF-SEC-IAM in the [SAP Support Portal](#). These contain information about program corrections and provide additional information.

To help you troubleshoot your issue, we also recommend increasing the log verbosity of your application and application router. We provide a [script](#) to help you. If for some reason you can't use this script, increase the log verbosity manually, see related link.

To troubleshoot problems with tokens from SAP Cloud Identity Services, see [Logging OpenID Connect Tokens](#) in the documentation for SAP Cloud Identity Services.

Our troubleshooting information can be found in [Troubleshooting for the SAP Authorization and Trust Management service in the Cloud Foundry environment](#). Check the individual troubleshooting topics for your error message. If you can't find your problem, create an incident in the component BC-CP-CF-SEC-IAM. For more information, see the related link.

- [The Security Tab Is Missing in the Subaccount](#)
- [Access Is Denied or Forbidden](#)
- [Identity Provider Could Not Process Authentication Request](#)
- [Logon Screen Shows "SAP HANA XS Advanced"](#)
- [Requested Route Does Not Exist](#)
- [Subdomain Does Not Map to a Valid Identity Zone](#)
- [No Client with Requested ID](#)
- [Login Issues](#)
- [Cannot Add Role Templates to Predefined Role Collections](#)
- [502 Error: Call to /oauth/token Was Not Successful](#)
- [Unexpected AuthnResponse : Existing authentication - <User>](#)
- [AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination>](#)
- [InResponseToField of Response doesn't correspond to the sent message](#)
- [Response issue time is either too old or with date in the future. Sync IdP to match skew <skew>](#)
- [Trust establishment issues](#)
- [Token retrieval fails with status code 401](#)
- [Cockpit displays HTTP status 500 error on logon with custom IdP user](#)
- [IAS application reference isn't created in your IAS tenant](#)
- [User base doesn't appear in existing Neo subaccounts](#)
- [User from corporate IdP can't log on to Neo subaccount](#)

- 409 error code on deleting a custom identity provider 
- Share Authorization and Trust Management service instances 
- Invalid "redirect\_uri" 

## Adding Authentication and Authorization

Developers create authorization information for business users in their environment and deploy this information in an application. They make this available to administrators, who complete the authorization setup and assign the authorizations to business users.

Developers store authorization information as design-time role templates in the `xs-security.json` security descriptor file. Using the cockpit, administrators of the environment assign the authorizations to business users.

The following sections contain the process of adding authentication and authorization checks for protecting your applications, links to a number of associated tutorials, extended tasks for creating authorization artifacts, as well as reference information, including the syntax required to set the properties and values defined in the application security descriptor file.

- [Protecting Your Application \[page 501\]](#)
- [Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#)
- [Application Security Descriptor Configuration Syntax \[page 545\]](#)

## Related Information

[Cloud Management Tools — Feature Set Overview \[page 129\]](#)

### 7.1.1 What Is the SAP Authorization and Trust Management Service?

Get a high-level overview of the concepts that underpin the SAP Authorization and Trust Management service for SAP BTP in the Cloud Foundry environment.

The SAP Authorization and Trust Management service lets you manage user authorizations and trust to identity providers. Identity providers are the user base for applications. We recommend that you use an SAP Cloud Identity Services tenant, an SAP on-premise system, or a custom corporate identity provider. User authorizations are managed using technical roles at the application level, which can be aggregated into business-level role collections for large-scale cloud scenarios.

## Environment

The SAP Authorization and Trust Management service is available for consumption in the following SAP BTP environments:

- Cloud Foundry
- Kyma
- Neo

### ⓘ Note

This documentation refers to SAP BTP for multi-environment subaccounts.

If you're looking for documentation about the Neo environment, see [Authorization and Trust Management in the Neo Environment](#).

## Features

### Use your corporate or a default IdP

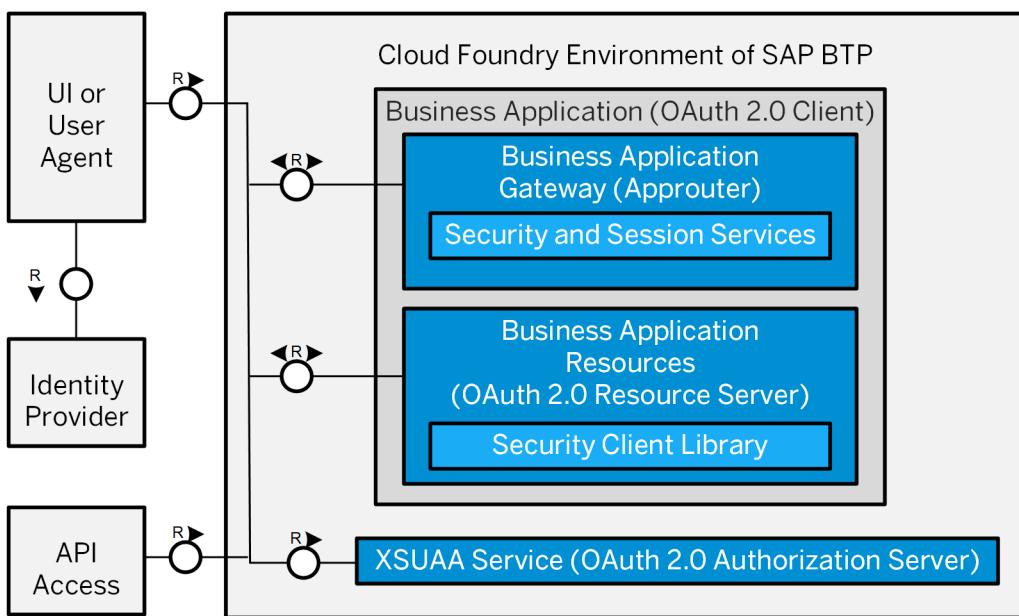
Enable user management for your applications by handling authentication to an external identity provider. Start with SAP ID service as a pre-configured easy-to-use identity provider. Switch to your corporate identity provider for customized user management.

### Enable role-based access to applications

Enable different privileges to users accessing your applications based on roles.

## Overview

The following figure shows a high-level overview of components, which comprise an SAP BTP business web applications and how these applications are embedded in the Cloud Foundry environment. Further details have been omitted for the sake of simplicity. It's further assumed that the Cloud Foundry environment is set up with basic configuration (that is, container-to-container networking isn't configured). The components and their interactions are depicted in the following block-diagram.



### **Application, Microservice, and App**

The components and their interactions are depicted in the previous block-diagram. It shows a runtime platform for business web applications. These are referred to as applications. An SAP BTP application is implemented in an architectural style that structures the application as a collection of loosely coupled components, termed microservices. Microservices can be deployed independently from one another. This eliminates the need to deploy the complete application if only a subset of its microservices have received new features or a bug fix. In the terminology of the Cloud Foundry environment of SAP BTP, microservices are referred to as apps.

### **Application Architecture**

The application consists of a distinct gateway app with at least one or more resource apps. The gateway serves as a reverse-proxy and provides functionality for security and session management. The application router app is a standard implementation of the gateway and is used as the single point-of-entry for the application. It also serves static content, initiates the authentication process, checks on cross-site request forgery (XSRF) attacks, and forwards requests to the resource apps while propagating user information.

The resource apps can use the security client library, which also provides security functionality. As stated in the previous section, all apps represent microservices, in the meaning of an app engineered and operated according to the 12-factor paradigm. All apps run in their dedicated runtime containers, which are hosted on the Cloud Foundry runtime platform.

### **OAuth 2.0, Resource Owner, Client, Resource Server, and the SAP Authorization and Trust Management Service**

The security functionality of SAP BTP is based on the OAuth 2.0 specification. OAuth 2.0 defines how a user - the OAuth 2.0 resource owner - can delegate all or a subset of the authorizations to a third-party application - the OAuth 2.0 client - without the third-party application needing to know the credentials of the user.

The Cloud Foundry environment uses a standard implementation of OAuth 2.0 to protect its platform resources (orgs, spaces, and platform operations on those entities).

However, the OAuth 2.0 specification is reused for SAP BTP with a proprietary implementation to protect the resources of business web applications powered by the Cloud Foundry environment. The proprietary implementation exchanges the responsibilities of the OAuth 2.0 entities, client and resource owner: the OAuth

2.0 client - represented by the application - holds all the authorizations. A set or sub-set of these authorizations is assigned to the user after authentication in the system. The application also acts as the OAuth 2.0 resource server because it contains the resource apps. All apps of an application operate under the same OAuth 2.0 client.

The SAP Authorization and Trust Management service (XSUAA) provides functionality for administrating and assigning application authorizations. It acts as the OAuth 2.0 authorization server and represents a typical reuse service. The SAP Authorization and Trust Management servicebroker creates a service instance for each application. Each app that wants to enforce authorizations with the security client library is then bound to this SAP Authorization and Trust Management serviceinstance of the corresponding application.

### Note

The Cloud Foundry environment also supports the following token grant types of Cloud Foundry.

- Authorization code grant
- Client credentials grant
- SAML 2.0 bearer grant

Refresh tokens are supported as well.

For more information, see the Cloud Foundry environment's API reference of the User Account and Authentication and the related link.

## **Service, Service Broker, and Service Instance**

A service is an app that includes service broker functionality. The service broker must implement the Open Service Broker API specification, for which the Cloud Foundry environment is a client. The service broker is responsible for advertising its service offerings and service plans to the Cloud Foundry environment of SAP BTP and acting on requests from the platform for provisioning, binding, unbinding, and deprovisioning.

A service instance represents a reserved resource and is an instantiation of a service offering for a service plan. The service offering is the advertisement of a service that the service broker supports. The service plan is a variant of the service offering and usually represents the costs and benefits of that plan.

The service broker binds the service instance to the consuming apps. The service binding contains information about the service (for example, URL, credentials), which the app uses to consume the service. Apps and services communicate with one another indirectly, using the Cloud Foundry router.

## **Reuse Service, Backing Service**

As a service is an app that includes service broker functionality, it can also run as a microservice component within an application. In this case, the service represents a reuse service. Services that don't represent components within applications, but rather run standalone, are referred to as backing services.

## **Authentication Against Trusted Parties Only**

The apps of the applications are accessed either using the UI of a user agent or the APIs that the app provides. All requests must first go through the Cloud Foundry router. Users of a user agent must first authenticate against a configured identity provider. The identity provider and the SAP Authorization and Trust Management service have a special trust relationship, which is established through crossover metadata configuration between the two systems (not depicted in the diagram above). Authentication is processed according to the SAML bearer assertion flow and initiated during the processing of the first request. A series of redirects leads to a request for authentication from the user agent towards the identity provider. After the user authenticates successfully, the identity provider responds with a SAML bearer assertion confirming the user's identity. This

SAML bearer assertion is presented to SAP Authorization and Trust Management service and the service determines the authorizations of that user. API clients receive their credentials directly from, and authenticate directly against, SAP Authorization and Trust Management service.

### JSON Web Tokens

A JSON web token (JWT) (according to RFC 7519) is an open standard that defines a compact token format for transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWT tokens can be signed using a secret key pair (with HMAC algorithm) or a public/private key pair using RSA.

The JWT token contains header and claims information (for example, issuer, subject, expiration time, consumer-defined information), and is digitally signed with the private key of the authorization server (UAA service).

The cloud or business application has a trust relationship with the authorization server. The trust is configured in the `<VCAP_SERVICES>` environment variable for the application router and each microservice of the business application. `<VCAP_SERVICES>` contains a credentials string for the UAA, which is created by the respective service broker when the application router and/or the microservice is bound to the UAA service instance. The credentials string contains, among other things, the public key corresponding to the private key of the UAA. This public key is used to verify the token signature.

### Related Information

<http://saml.xml.org/saml-specifications> ↗

<https://tools.ietf.org/html/rfc6749#section-1.4> ↗

[SAP Authorization and Trust Management Service \[page 3023\]](#)

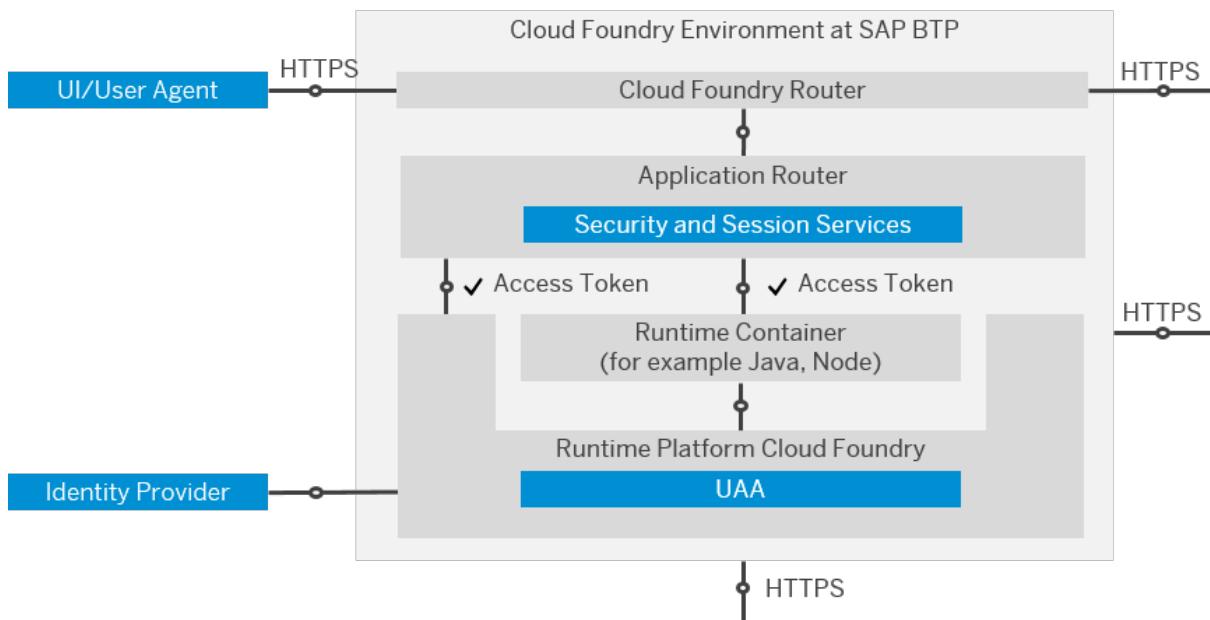
[Cloud Foundry API Reference for User Account and Authentication Server API](#) ↗

[Security in the Kyma Environment \[page 3120\]](#)

## 7.1.2 Web Access Control

The Cloud Foundry environment extends SAP BTP. It provides platform security functions such as business user authentication, authorization management, and other security functions for access to the applications in the runtime container. To access the runtime container, the business user can use a browser or a browser-based user interface.

The following diagram shows the architecture with the components that are responsible for business user authentication, authorization management, and security. It is not mandatory for applications to use the User Account and Authentication service and the application router.



The User Account and Authentication (UAA) component provides a programming model for business applications. It is the central infrastructure component of the runtime platform for business user authentication and authorization management. The users can be stored in the following identity providers:

- SAP ID service
- SAP Cloud Identity Services
- Any SAML 2.0 identity provider

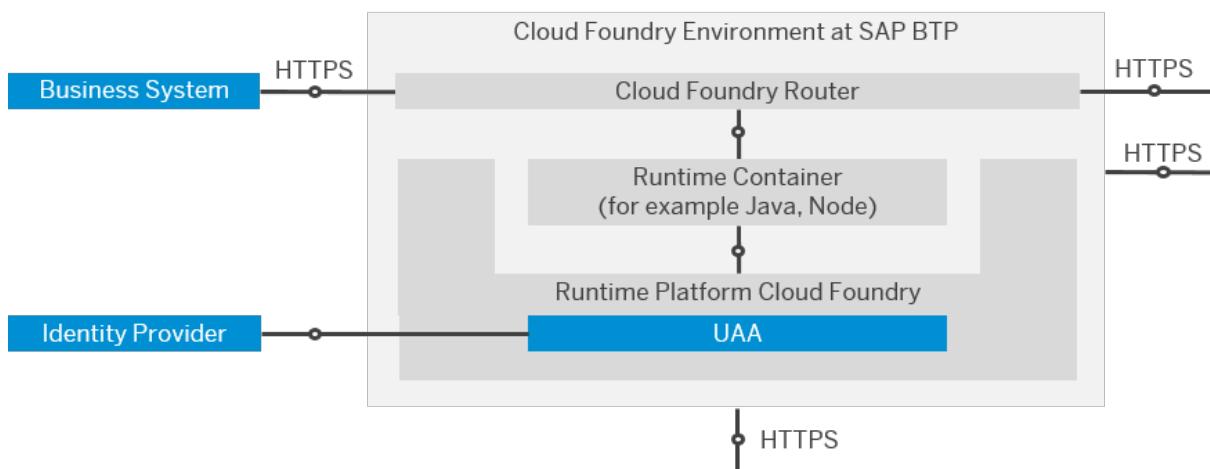
Applications authenticate using OAuth 2.0. When business users access an application, the application router acts as OAuth client and redirects their request to the OAuth authorization server for authentication (see the [Applications](#) section). Runtime containers act as resource servers, using the container security API of the relevant container (for example, Java) to validate the token issued by the OAuth authorization server.

## 7.1.3 API Access Control

The Cloud Foundry environment extends SAP BTP. It provides platform security functions such as business system authentication, authorization management, and other security functions to enable business systems to access the applications (for example, Java or Node.js) in the runtime container. Business systems use APIs to access the runtime container.

A business system uses APIs to directly access the resources in the runtime container.

The following diagram shows the architecture with the components that are responsible for business system authentication, authorization management, and security.



The User Account and Authentication (UAA) component is the central infrastructure component of the runtime platform for authentication and authorization management. The users can be stored in the following identity providers:

- SAP Cloud Identity Services
- Any identity provider

Business system use APIs to directly access the resources in the runtime container. The UAA acts as an OAuth authorization server and issues an appropriate access token. It enables the business system to directly access an application in the runtime container. Runtime containers act as OAuth resource servers, using the container security API of the relevant container (for example, Java) to validate the token issued by the OAuth authorization server.

## Related Information

[Protecting Your Application \[page 501\]](#)

[Tutorials for the SAP Authorization and Trust Management Service \[page 527\]](#)

## 7.1.4 Authorization Entities

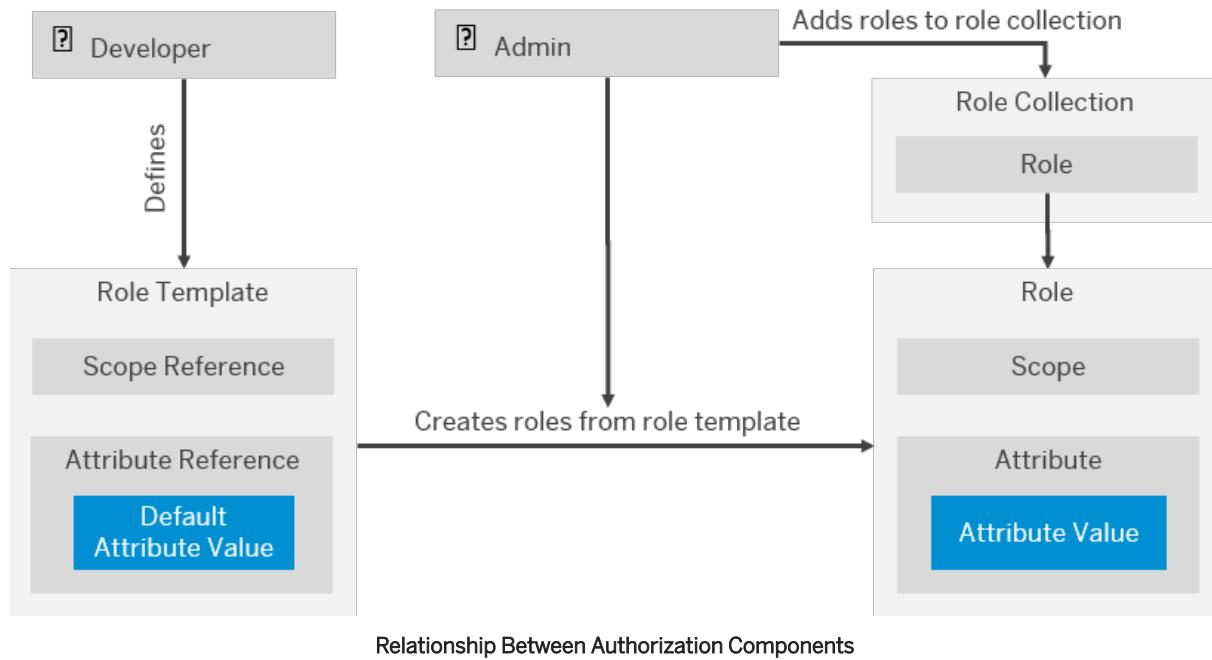
Business users in an application require different authorizations because they work in different jobs.

For example, in a leave request process, there are employees who want to create and submit leave requests, managers who approve or reject, and payroll administrators who need to see all approved leave requests to calculate the leave reserve. The authorization concept of a leave request application has to cover the needs of these employee groups. This authorization concept includes elements such as roles, scopes, and attributes.

A role is an instance of a role template; you can build a role based on a role template and assign the role to a role collection.

Role templates refer to attributes and scopes. The role templates contain the authorizations for activities, such as viewing, editing, or deleting data.

Information that is specific to the user is stored in attributes. For each attribute, administrators can specify the value that restricts data access. Static attributes are stored in the role, whereas in cases in which a custom identity provider provides the business users, you can dynamically reference all the attributes that come with the access token.



## Scopes for Functional Authorization Checks

The application defines scopes, which describe the authorizations required by the application to execute functions such as editing, viewing, and deleting.

The application security descriptor file (`xs-security.json`) defines the scopes used by the application.

## Role Templates

A role template is a description of one or more roles (for example, employee or manager) and any attributes that apply to those roles.

If a role template contains attributes, the administrator must create different versions of the roles and fill in the attributes that are subject to customization; for example, one where country equals UK and another where country equals USA. This instantiates the role template. Else, the roles will have empty attributes (with no concrete values).

Role templates that contain only application-specific local scopes can be instantiated without the administrator having to do anything.

## Role Collections

Role collections reference role templates.

After the developer has created the role templates and deployed them to the relevant application, it's the administrator's task to use those role templates to build roles with the required attributes, aggregate the roles into role collections, and then assign the role collections to business users in the application.

## Attributes

Information retrieved from the user's identity (such as department or cost center) is stored in attributes. Attributes refine the authorizations of business users according to the attributes that come with the business users. For example, you can use static attribute values such as country equals USA, or dynamic attribute values from the access token or identity provider, such as group ID.

## 7.1.5 Monitoring and Troubleshooting

This section provides information on troubleshooting-related activities for the SAP Authorization and Trust Management service in the Cloud Foundry environment.

### → Tip

We also recommend that you regularly check the SAP Notes and Knowledge Base for component BC-CP-CF-SEC-IAM in the [SAP Support Portal](#). These contain information about program corrections and provide additional information.

To help you troubleshoot your issue, we also recommend increasing the log verbosity of your application and application router. We provide a [script](#) to help you. If for some reason you can't use this script, increase the log verbosity manually, see related link.

To troubleshoot problems with tokens from SAP Cloud Identity Services, see [Logging OpenID Connect Tokens](#) in the documentation for SAP Cloud Identity Services.

Our troubleshooting information can be found in [Troubleshooting for the SAP Authorization and Trust Management service in the Cloud Foundry environment](#). Check the individual troubleshooting topics for your error message. If you can't find your problem, create an incident in the component BC-CP-CF-SEC-IAM. For more information, see the related link.

- [The Security Tab Is Missing in the Subaccount](#)
- [Access Is Denied or Forbidden](#)
- [Identity Provider Could Not Process Authentication Request](#)
- [Logon Screen Shows "SAP HANA XS Advanced"](#)
- [Requested Route Does Not Exist](#)
- [Subdomain Does Not Map to a Valid Identity Zone](#)
- [No Client with Requested ID](#)

- Login Issues 
- Cannot Add Role Templates to Predefined Role Collections 
- 502 Error: Call to /oauth/token Was Not Successful 
- Unexpected AuthnResponse : Existing authentication - <User> 
- AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination> 
- InResponseToField of Response doesn't correspond to the sent message 
- Response issue time is either too old or with date in the future. Sync IdP to match skew <skew> 
- Trust establishment issues 
- Token retrieval fails with status code 401 
- Cockpit displays HTTP status 500 error on logon with custom IdP user 
- IAS application reference isn't created in your IAS tenant 
- User base doesn't appear in existing Neo subaccounts 
- User from corporate IdP can't log on to Neo subaccount 
- 409 error code on deleting a custom identity provider 
- Share Authorization and Trust Management service instances 
- Invalid "redirect\_uri" 

## Related Information

[Getting Support \[page 3148\]](#)

[Enable and Provide Application Logs \[page 3039\]](#)

[Auditing and Logging Information for SAP Authorization and Trust Management Service \[page 3042\]](#)

[Authorization and Access Control Cookbook !\[\]\(62bf6b6541ff61819bc8e9239af98907\_img.jpg\)](#)

### 7.1.5.1 Enable and Provide Application Logs

If there are authentication problems in your application, enable logging for the container security library in question, reproduce the problem, and attach the application logs. To obtain more details, set the environment variables for the application.

## Context

→ Tip

We also provide a [log collector script !\[\]\(fbd1200cf91c98304ba5ce9a0d72ca02\_img.jpg\)](#), that helps you to increase the log verbosity of your application and application router.

## Procedure

1. Open a command prompt.
2. Log on to your Cloud Foundry environment using the Cloud Foundry command line interface (CLI). For more information, see [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 2453\]](#).
3. Choose your organization.

### ⓘ Note

The Cloud Foundry command line interface prompts you to choose an org. To find the org of your subaccount, use the SAP BTP cockpit to go to your subaccount. You find the org in the Cloud Foundry tile under *Organization* (see [Navigate to Orgs and Spaces \[page 2433\]](#)).

4. Choose the space where the application is located.
5. To route the log messages to the standard output, use the following command:

```
cf set-env <application_name> SAP_EXT_TRC stdout
```

### ⚡ Example

```
cf set-env your-app SAP_EXT_TRC stdout
```

6. To set the log level, use the following command:

```
cf set-env <application_name> SAP_EXT_TRL 3
```

### ⚡ Example

```
cf set-env your-app SAP_EXT_TRL 3
```

7. (For Node.js) To set detailed logs of the Security API for Node.js, use the following command:

```
cf set-env <application_name> DEBUG xssec*
```

### ⚡ Example

```
cf set-env your-app DEBUG xssec*
```

8. Restage your application using the following command:

```
cf restage <application>
```

### ⚡ Example

```
cf restage your-app
```

Restaging the application enables the display of the logs.

9. We recommend piping the output to a log file. Use the following command to do this:

```
cf logs <application> > <log_file_name>
```

### ⚡ Example

```
cf logs your-app > your-app-log.txt
```

10. Reproduce the problem in your application. The events that occur in your application are logged in your local log file.

#### ⓘ Note

You can stop the recording of the log messages using `[CTRL+C]`. You can revert the environment variables using the following command:

```
cf unset-env <application> SAP_EXT_TRC
```

#### ⚡ Example

```
cf unset-env your-app SAP_EXT_TRC
```

Restage your application using the following command:

```
cf restage <application>
```

#### ⚡ Example

```
cf restage your-app
```

11. Create an incident for your local support. Create an incident using the component BC-CP-CF-SEC-IAM. Use the [SAP Support Portal](#). For more information, see [Monitoring and Troubleshooting \[page 3038\]](#).
12. Attach the log files to the incident.

To obtain more details about the token validation, set the following environment variables of the container security library and Node.js (if applicable):

SAPSSEXT Environment Variables

Environment Variables	Description
<code>SAP_EXT_TRC</code>	Enablement of logs
<code>SAP_EXT_TRL</code>	Log level ranging from off to high Values (integer): <ul style="list-style-type: none"><li>• <code>0</code> (off)</li><li>• <code>1</code> (default)</li><li>• <code>2</code> (medium)</li><li>• <code>3</code> (high)</li></ul>

Node.js Environment Variables

Environment Variables	Description
<code>DEBUG</code>	Detailed logs of the security API for Node.js Value (string): <code>xsssec*</code>

## 7.1.5.2 Auditing and Logging Information for SAP Authorization and Trust Management Service

Here you can find a list of the security events that are logged by SAP Authorization and Trust Management service (XSUAA). These events are provided in addition to the events of the Cloud Foundry User Account and Authentication service (UAA).

### Security Events of the User Account and Authentication Service

For information about the security events of the UAA, see [UAA Audit Requirements](#) in the Cloud Foundry documentation.

### Security Events of the SAP Authorization and Trust Management Service

The SAP Authorization and Trust Management service uses role collections to handle account management.

SAP Authorization and Trust Management service records all its changes in its database tables and summarizes these changes in the audit log. The following table summarizes the audit log entries.

#### ⓘ Note

UNKNOWN\_USER in an audit log message is a placeholder, which usually appears when logs are written for flows that don't involve users, for example, client credential flows.

#### Security Events Written in Audit Logs

Event grouping	What events are logged	How to identify related log events	Additional information
Identity provider management	Trust identity provider.	object with type "IdentityProvider" and id consisting of: crudType "CREATE"	Attributes of the identity provider.
		tableName "xs_tenant", crudType "UPDATE"	Trusting an identity provider using OpenID Connect triggers a change in the XSUAA tenant.

Event grouping	What events are logged	How to identify related log events	Additional information
	Update trust with identity provider.	object with type "IdentityProvider" and id consisting of: crudType "UPDATE"	Old attributes of the identity provider and any attributes required to identify the changes.
		tableName "xs_tenant", crudType "UPDATE"	Updating trust in an identity provider using OpenID Connect can trigger a change in the XSUAA tenant.
	Delete trust with identity provider.	object with type "IdentityProvider" and id consisting of: crudType "DELETE"	Attributes of the service instance.
	Create instance.	tableName "xs_tenant", crudType "UPDATE" Attribute with name "complete" and value " "ServiceInstanceId : <Instance_ID> ...	Removing trust in an identity provider using OpenID Connect triggers a change in the XSUAA tenant.
		object with type "<Instance_Name>" and id consisting of: crudType "CREATE"	Attributes of the service instance.

Event grouping	What events are logged	How to identify related log events	Additional information						
	Update instance.	<pre>Attribute with name "complete" and value " "ServiceInstanceId : &lt;Instance_ID&gt; ... object with type "&lt;Instance_Name&gt;" and id consisting of: crudType "UPDATE"</pre>	Old attributes of the service instance and any attributes required to identify the changes.						
	Delete instance.	<pre>Attribute with name "complete" and value " "ServiceInstanceId : &lt;Instance_ID&gt; ... object with type "&lt;Instance_Name&gt;" and id consisting of: crudType "DELETE"</pre>	Attributes of the service instance.						
<p><b>① Note</b></p> <p>If you create a new binding secret for the service instance, a new binding entry is created in the SAP Authorization and Trust Management service. This entry then appears in the audit log of the subaccount of the service instance. The audit log of the subaccount of the reuse service itself isn't affected.</p>									
Role collection management	Create role collection.	<table border="0"> <tr> <td data-bbox="795 1563 933 1608">tableName</td> <td data-bbox="1081 1563 1271 1608">Other attributes:</td> </tr> <tr> <td data-bbox="795 1612 1065 1680">"xsrolecollections" , crudType "CREATE"</td> <td data-bbox="1081 1612 1351 1754"> <ul style="list-style-type: none"> <li>• Timestamp</li> <li>• Origin key</li> <li>• Role collection name</li> <li>• Zone ID</li> </ul> </td> </tr> <tr> <td data-bbox="795 1702 1065 1747">audit.configuration</td> <td></td> </tr> </table>	tableName	Other attributes:	"xsrolecollections" , crudType "CREATE"	<ul style="list-style-type: none"> <li>• Timestamp</li> <li>• Origin key</li> <li>• Role collection name</li> <li>• Zone ID</li> </ul>	audit.configuration		
tableName	Other attributes:								
"xsrolecollections" , crudType "CREATE"	<ul style="list-style-type: none"> <li>• Timestamp</li> <li>• Origin key</li> <li>• Role collection name</li> <li>• Zone ID</li> </ul>								
audit.configuration									

Event grouping	What events are logged	How to identify related log events	Additional information
	Assign users to role collection.	Direct assignment to users: tableName "xs_rolecollection2user", crudType "CREATE"  Mapping to users with attributes: tableName "xsrolecollection2samlattribute", crudType "CREATE" audit.configuration	Other attributes for direct assignment: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• User SCIM ID</li><li>• Zone ID</li><li>• Role collection name</li></ul> Other attributes for mapping: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• Attribute name</li><li>• Attribute value</li><li>• Identity provider URL</li><li>• Zone ID</li><li>• Role collection name</li></ul>
	Assign roles to role collection.	tableName "xsrolecollection2rrole", crudType "CREATE"	<ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• Role name</li><li>• Role zone ID</li><li>• Role template name</li><li>• Role collection zone ID</li><li>• Role template app ID</li><li>• Role collection name</li></ul>
	Delete role collection.	tableName "xsrolecollections", crudType "DELETE" audit.configuration	Other attributes: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• Role collection name</li><li>• Zone ID</li></ul>
Role management	Create role.	tableName "xsrole", crudType "CREATE" audit.configuration	Other attributes: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• App ID</li><li>• Role name</li><li>• Role template name</li><li>• Zone ID</li></ul>

Event grouping	What events are logged	How to identify related log events	Additional information
	Modify attribute values in role.	tableName "xsattribute2role", crudType "CREATE" audit.configuration	Other attributes: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• Role name</li><li>• Zone ID</li><li>• Attribute app ID</li><li>• Role template name</li><li>• Attribute name</li><li>• Attribute value</li><li>• Role template app ID</li></ul>
	Delete role.	tableName "xsrole", crudType "DELETE" audit.configuration	Other attributes: <ul style="list-style-type: none"><li>• Timestamp</li><li>• Origin key</li><li>• Role template app ID</li><li>• Role name</li><li>• Role template name</li><li>• Zone ID</li></ul>
Emergency administrator assignment	Assign global account or subaccount administrator in case of an emergency	<p>"crudType": "CREATE", "type": "Global Account Administrator assignment"</p> <p>or</p> <p>"crudType": "CREATE", "type": "Subaccount Administrator assignment"</p> <p>audit.security-events</p>	<p>The user who assigned the permission is identified by the user name and by the origin key (such as sap.default for the default identity provider).</p> <p>Further information included in the log:</p> <ul style="list-style-type: none"><li>• Initiating user name</li><li>• Initiating user origin</li><li>• New administrator's user name</li><li>• New administrator's origin</li></ul>

#### ❖ Example

- Assignment of a subaccount administrator by a global account administrator

Event grouping	What events are logged	How to identify related log events	Additional information
Token Embedding	Embedding error	Error during handling of IAS Tokens for embedding.  audit.security-events	Occurs when an error occurs when attempting to exchange a token for a token with an embedded token from SAP Cloud Identity Services or a corporate identity provider. Check the configuration of the application.
			For more information, see <a href="#">Include Tokens from Corporate Identity Providers or SAP Cloud Identity Services in Tokens of the SAP Authorization and Trust Management Service [page 540]</a> .
SAML authentication	Authentication error	SAMLAutenticationError  Response issue time is either too old or with date in the future. Sync IdP to match skew <skew>  audit.security-events	Occurs when the time skew between SAP Authorization and Trust Management service and the identity provider is larger than 60 seconds. Or the authentication response took more than 60 seconds to reach the SAP Authorization and Trust Management service after being issued.
			Check the time skew between the identity provider and SAP Authorization and Trust Management service. Synchronize the clock of the identity provider.

Event grouping	What events are logged	How to identify related log events	Additional information
		<pre>SAMLAuthenticationE rror  Unexpected AuthnResponse : Existing authentication - &lt;user&gt;  audit.security-events</pre>	<p>The user has probably chosen the back button on the browser, triggering a second authentication request to the identity provider with the same user ID. The identity provider issues a second authentication response for the same user ID. SAP Authorization and Trust Management service rejects duplicate responses.</p>
		<pre>SAMLAuthenticationE rror  AuthnRequest expired - ID: &lt;request_id&gt; Destination: &lt;identity_provider_&lt;br/&gt;destination&gt;  audit.security-events</pre>	<p>Occurs when an authentication response from an identity provider takes more than 15 minutes.</p> <p>If this error occurs consistently, check why the identity provider needs more than 15 minutes to issue an authentication response.</p>
		<pre>SAMLAuthenticationE rror  InResponseToField of Response doesn't correspond to the sent message  audit.security-events</pre>	<p>Occurs when a user attempts to log on or refresh a session for which the authentication request has expired, for example, if this message is preceded by AuthnRequest expired - ID.</p>
		<pre>SAMLAuthenticationE rror  No valid credential to evaluate the token  audit.security-events</pre>	<p>Occurs when the certificate used to sign the SAML response isn't valid.</p>

Event grouping	What events are logged	How to identify related log events	Additional information
	Authentication success	UserAuthenticationSuccess audit.security-events	<p>These entries are in addition to the entries made by the UAA. See the previous section <i>Security Events of the User Account and Authentication Service</i>. Authentication success includes:</p> <ul style="list-style-type: none"> <li>• User name</li> <li>• Principle (SCIM user ID)</li> <li>• Origin key</li> <li>• Zone ID</li> </ul>
	SAML responses	<pre>"msgNo":&lt;index&gt;, "msgId": "&lt;message_id&gt;"</pre> audit.security-events	<p>We include SAML responses in the audit log for web single sign-on and SAML bearer assertions.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <b>① Note</b>  <p>When messages exceed 4k, we break the messages into multiple entries. We identify each message with a msgId GUID and the parts with a msgNo index. To view the whole SAML response, gather the parts and stitch the contents together.</p> </div>

### 7.1.5.3 Troubleshooting for SAP Authorization and Trust Management Service

This section contains troubleshooting information for the Cloud Foundry environment of SAP Business Technology Platform. It provides assistance with issues concerning platform security functions such as business user authentication, authentication of applications, authorization management, trust management, and other security functions.

[The Security Tab is Missing in the Subaccount \[page 3050\]](#)

[Access Is Denied or Forbidden \[page 3051\]](#)

[Identity Provider Could Not Process Authentication Request \[page 3052\]](#)

[Logon Screen Shows "SAP HANA XS Advanced" \[page 3053\]](#)  
[Requested Route Does Not Exist \[page 3055\]](#)  
[Subdomain Does Not Map to a Valid Identity Zone \[page 3056\]](#)  
[No Client with Requested ID \[page 3058\]](#)  
[Login Issues \[page 3059\]](#)  
[Cannot Add Role Templates to Predefined Role Collections \[page 3060\]](#)  
[409 Error Code on Deleting a Custom Identity Provider \[page 3061\]](#)  
[Unexpected AuthnResponse : Existing authentication - <User> \[page 3062\]](#)  
[AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination> \[page 3063\]](#)  
[InResponseToField of Response Doesn't correspond to the Sent Message \[page 3063\]](#)  
[Response Issue Time is Either too Old or with Date in the Future. Sync IdP to Match Skew <skew> \[page 3064\]](#)  
[Trust Establishment Issues \[page 3065\]](#)  
[Token Retrieval Fails With Status Code 401 \[page 3066\]](#)  
[Cockpit Displays HTTP Status 500 Error on Logon with Custom IdP User \[page 3068\]](#)  
[IAS Application Reference isn't Created in Your IAS Tenant \[page 3069\]](#)  
[User Base Doesn't Appear in Existing Neo Subaccounts \[page 3070\]](#)  
[User from Corporate IdP Cannot Log on to Neo Subaccount \[page 3070\]](#)  
[Share Authorization and Trust Management service instances \[page 3071\]](#)  
[Invalid "redirect\\_uri" \[page 3072\]](#)  
[400 Error: Call to /oauth/token Was Not Successful \[page 3073\]](#)

### 7.1.5.3.1 The Security Tab is Missing in the Subaccount

#### Symptom

If you navigate to your subaccount in the SAP BTP Cockpit, you can't see the Security tab.

#### Reason and Prerequisites

The problem here is that only the user that created the subaccount is granted the security administrator role. If other subaccount users should be able to see the security tab and make changes to the security configuration, the creator of the subaccount has to grant them the security administrator role.

The creator of the subaccount and the users that should be added must have at least one of the following member roles:

- They are members of the Cloud Foundry organization (if available) in the subaccount.
- They are members of any Cloud Foundry space that belongs to the organization.
- They are members of the global account that contains your subaccount.

## Solution

The creator of the subaccount has to grant the security administrator role. To do this, follow these steps:

1. Open your SAP BTP Cockpit.
2. Navigate to your subaccount.
3. Choose the **Security** tab and choose **Administrators**.
4. Choose **Add Administrators**.  
A popup opens, where you can enter the user ID and assign roles.
5. Type in the ID (e-mail) of the user, that you want to add as a security administrator.
6. To assign roles, make sure that the **User & Role Administrator** checkbox is selected.
7. Choose **OK**.

After the role has been granted, the user needs to log out and log back in again to make sure the changes are applied to their subaccount user.

### 7.1.5.3.2 Access Is Denied or Forbidden

#### Symptom

After sending a request to a web application in the SAP BTP, Cloud Foundry environment, you get a response containing a nested error element with the value access\_denied and an error\_description element.

The system responds with Access is denied.

The system might also respond with the single word Forbidden.

#### ↔ Output Code

```
<oauth>
  <error_description>Access is denied</error_description>
  <error>Access denied</error>
</oauth>
```

#### Reason and Prerequisites

A business user has tried to access the URL endpoint of a web application but is not authorized to do this. The authorizations (scopes) required for the URL endpoint must be assigned to this business user. Scopes

(functional authorizations) allow the business user to perform operations provided by the web application. Attributes (instance-based authorizations) specify the data, for example, company code or cost center, which the business user is authorized to use. In this way, business users can perform the operations they are authorized for.

## Solution

A role collection with the role that contains the required authorizations must be available. Assign this role collection to the business user.

If there are neither instance-based authorizations nor roles containing the required attribute values, create a role based on the corresponding role template. Maintain the attribute values of the new role to define the required data authorizations.

### Note

- Role templates are predefined and deployed together with the web application. Role templates specify the scopes (functional operations) that the web application supports. You cannot change existing role templates or create new ones.
- After a role collection has been assigned, you might need to clear the cache and cookies and log on to the application again for the changes to take effect.

For more information about roles and role collections, see [Building Roles and Role Collections for Applications](#).

### 7.1.5.3.3 Identity Provider Could Not Process Authentication Request

#### Symptom

After sending a request to a web application in the SAP BTP, Cloud Foundry environment, the system responds as follows:

Error - Identity provider could not process the authentication request received.  
Delete your browser cache and stored cookies, and restart your browser. If you continue to experience issues, send an e-mail to [sso@sap.com](mailto:sso@sap.com).

#### Reason and Prerequisites

The Cloud Foundry environment uses the Identity Authentication service as its SAML 2.0 identity provider. The SAP Authorization and Trust Management service (XSUAA) forwards unauthenticated requests to the

configured identity provider. This error occurs because the trust relationship between the SAP Authorization and Trust Management service and the identity provider hasn't been fully configured. The subaccount has not been registered in the SAML 2.0 identity provider.

## Solution

Configure the missing part of the trust relationship in your identity provider by doing one of the following:

- Use the [simplified trust configuration](#).
- Download the SAML metadata file from the SAP Authorization and Trust Management service, upload it to your SAML 2.0 identity provider, and complete the configuration by registering your subaccount. The following link provides instructions for configuring the Identity Authentication service as a trusted SAML 2.0 identity provider: [Register SAP BTP Subaccount in the SAML 2.0 Identity Provider](#).

### 7.1.5.3.4 Logon Screen Shows "SAP HANA XS Advanced"

#### Symptom

After sending a request to a web application in the SAP BTP, Cloud Foundry environment, you see a logon screen with the title SAP HANA XS Advanced.

Below the Log On field, the following domain is displayed: xs2security.accounts400.ondemand.com

ⓘ Note

The expected system behavior is to return a logon screen with the title Welcome <IdP\_tenant\_name>!.

#### Reason and Prerequisites

The request of the business user is not authenticated, so it needs to be redirected to the identity provider where the business user is defined. The trust of the identity provider is configured and assigned to the tenant to which the business user belongs. In this case, SAP BTP, Cloud Foundry environment cannot correctly determine the tenant of the business user. As a result, it cannot find the correct identity provider and falls back to

<https://authentication.sap.hana.ondemand.com>

## Solution

This error indicates a problem with the `TENANT_HOST_PATTERN` environment variable in the `manifest.yml` file of the web application. The tenant host pattern is a regular expression (regex) and is used to determine the tenant ID (which is identical to the subdomain) with which the request is associated. For more information, see [Multitenancy](#). The system shows the behavior described earlier if the URL of the request does not match with the defined pattern.

The tenant host pattern should be defined as follows:

```
.
.
.
env:
  XSAPPNAME: <application_name>
  TENANT_HOST_PATTERN: "^(.*)-<application-router-
hostname>.cfapps.<region>.hana.ondemand.com".
```

### ⓘ Note

Separate the regular expression and the Cloud Foundry host name with a hyphen (-). The subdomain of the tenant is not a subdomain of the web application in a technical sense. The subdomain would be separated from the Cloud Foundry host name with a dot (.). Since there is no automatic certificate creation process each time that a new tenant is onboarded, you must use a workaround.

The current workaround is to concatenate the subdomain of the tenant with the Cloud Foundry host name using the hyphen (-). All such concatenations share the same Cloud Foundry top level domain, together with the certificate of this top-level domain.

The following example shows the web application

`bulletinboard`

in the

`cfapps.eu10.hana.ondemand.com`

Cloud Foundry domain:

```
.
.
.
env:
  XSAPPNAME: bulletinboard
  TENANT_HOST_PATTERN: "^(.*)-approuter.cfapps.eu10.hana.ondemand.com".
```

### ⓘ Note

Recheck the definition of the `TENANT_HOST_PATTERN` environment variable in the `manifest.yml` file, and correct it, if necessary. **The route for the `TENANT_HOST_PATTERN` has to be lowercase.**

## 7.1.5.3.5 Requested Route Does Not Exist

### Symptom

You send a request to a web application, which is running on SAP BTP, Cloud Foundry environment and the browser responds with the message:

```
404 Not Found: Requested route ('<cloudFoundryHostname>.<cloudFoundryDomain>') does not exist.
```

### Reason and Prerequisites

The URL of the request contains a route to an application for which no route mapping has been defined. This means that the Cloud Foundry router can't delegate the request to the respective web application and responds with an error message. Another reason might be that your application has stopped.

### Solution

#### Create and Map the Missing Route to the Web Application

Create and map the missing route to the web application.

```
cf map-route <webApplication> <cloudFoundryDomain> --hostname <cloudFoundryHostName>
```

##### Example

The web application's name is bulletinboard-123456 and the application was created in the European landscape eu10.hana.ondemand.com, the application should be reachable under the following domain: <https://bulletinboard.eu10.hana.ondemand.com>.

1. Use the following command to create this route:  

```
cf map-route bulletinboard-123456 cfapps.eu10.hana.ondemand.com --hostname bulletinboard
```
2. Check if the route was created:  

```
cf routes
```

#### Check if Your Application is Still Running and Restart it

1. Log in to your Cloud Foundry account.  

```
cf routes
```
2. Choose your ORG and SPACE.
3. Check the status of your application.  

```
cf app <your application>
```

4. Restart your application.  

```
cf restart <your application>
```
5. Your application should now have the status running and you can call it through its route.

## 7.1.5.3.6 Subdomain Does Not Map to a Valid Identity Zone

### Symptom

You send an unauthenticated request to a web application, which is running on SAP BTP, Cloud Foundry. The system responds with the message:

The subdomain does not map to a valid identity zone.

### Reason and Prerequisites

The subdomain and/or the Cloud Foundry domain of the web application do not match the values provided in the URL of the web application.

Reason	Solution
The URL contains either an incorrect subdomain or an incorrect landscape domain (or both).	Check the subdomain and landscape domain of your request
The subdomain belongs to a subaccount which has not yet been defined in the Cloud Foundry environment.	Create the missing subaccount
The TENANT_HOST_PATTERN in the manifest.yml file of the web application is incorrect.	Correct the TENANT_HOST_PATTERN in the manifest.yml file of the web application
You are running your applications on the SAP BTP trial and your account has expired.	Extend your trial account

### Solution

#### Check the Subdomain and Landscape Domain of Your Request

Check the subdomain and landscape domain of your request. You can find the subdomain in the SAP BTP cockpit in the **Overview** section of the subaccount that you are using for the request, under **Subaccount Details**.

You can find the landscape URL for your Cloud Foundry environment [here](#). For example, the subdomain for the European region is **eu10.hana.ondemand.com**.

#### Create the Missing Subaccount

Create the missing subaccount with the same subdomain in the Cloud Foundry environment of the respective Cloud Foundry domain.

## Correct the TENANT\_HOST\_PATTERN in the manifest.yml file of the web application

1. Check the TENANT\_HOST\_PATTERN environment variable in the `manifest.yml` file of the web application. The tenant host pattern is a regular expression (regex) and is used to determine the subdomain that the request is associated with. If the URL of the request does not match the pattern in the `manifest.yml`, you will receive an error message.
2. The tenant host pattern should usually be defined as follows:

```
env:  
  XSAPPNAME: "<your-application-name>"  
  TENANT_HOST_PATTERN: "^(.*)-<CloudFoundryHostName>.<landscape>".
```

The regular expression and the Cloud Foundry hostname are separated by a hyphen. The subdomain is not a subdomain of the web application in a technical sense. The subdomain would otherwise be separated from the Cloud Foundry hostname with a dot. That, however, would require the automation of the certificate creation process each time a new tenant is onboarded. The workaround for the time being is therefore to concatenate the subdomain of the tenant with the Cloud Foundry hostname with a hyphen. All of these concatenations share the same Cloud Foundry top level domain and also the certificate of that top level domain.

### ⌚ Example

The web application's name is **bulletinboard-123456** and the application was created in the European landscape **eu10.hana.ondemand.com**.

1. Change the `manifest.yml`.

```
env:  
  XSAPPNAME: "bulletinboard-123456"  
  TENANT_HOST_PATTERN: "^(.*)-  
  approuter-123456.cfapps.eu10.hana.ondemand.com".
```

2. Recheck the definition of the TENANT\_HOST\_PATTERN environment variable in the `manifest.yml` file and correct it, if necessary.

### ⓘ Note

The route for the TENANT\_HOST\_PATTERN has to be lowercase.

## Extend your trial account

Extend your trial account from within the SAP BTP cockpit.

1. Visit the [trial page](#).
2. Choose **Enter Your Trial Account**.
3. Choose the calendar symbol in the upper right corner.
4. Choose **Extend Free Trial**.

Cloud Foundry trial accounts expire after 30 days. You can extend the trial period to a maximum of 90 days, after which your account is automatically deleted.

## 7.1.5.3.7 No Client with Requested ID

### Symptom

You are not yet authenticated, and you send your first request to a web application, which is running on SAP BTP, Cloud Foundry environment. After you enter your credentials in the login screen, the system responds with the following error message:

```
No client with requested id: <OAuth2 Client-Id> - Authorization Request Error -  
There was an error. The request for authorization was invalid.
```

The URL in the address-field of the browser contains the following URL-pattern `https://<subdomain>.authentication.<cloudFoundryDomain>/oauth/authorize?<urlParameters>`.

### Reason and Prerequisites

#### Technical Reason

Even though you were authenticated successfully, **the SAP Authorization and Trust Management service (XSUAA) could not assign the authorizations** (scopes and attributes) of your business user to the web application that you initially called. **The web application acts towards the SAP Authorization and Trust Management service as the OAuth2 client** and recognizes if the business user is sending an authorized request or not. If the request is not authorized, the web application first redirects the request to the authorized endpoint of the SAP Authorization and Trust Management service.

**The SAP Authorization and Trust Management service acts towards the web application as the OAuth2 Authorization server** and recognizes if the request to be authorized is authenticated or not. If the request is not authenticated, the service redirects the request to the configured identity provider (IdP) to initiate the authentication process.

The IdP sends a login screen to your browser so that you can authenticate yourself. The IdP then redirects your answer then back to the service with a SAML2 Bearer Assertion, which holds the authentication information of your business user. The SAP Authorization and Trust Management service can now retrieve the authorizations of your business user.

The service now wants to assign these authorizations to the OAuth2 client (the web application). The web application provides its OAuth2 client-id in the `client_id` URL parameter of the authorized endpoint. **The main point to consider here is that the SAP Authorization and Trust Management service accepts only the client-ids of those OAuth2 clients that it already knows.** That means that OAuth2 clients must register themselves to the service, before they call the authorized endpoint. This error occurs because **the web application was not registered as a valid OAuth2 client for the service**, before it called the authorized endpoint.

#### Reasons

Reason	Solution
--------	----------

You didn't create a service instance of the SAP Authorization and Trust Management service for your application. Create a service instance for your application.

---

You're trying to access a SaaS application that you have unsubscribed from. Subscribe to the SaaS application.

---

## Solution

### Create a service instance for your own application

1. Create a service instance in the space in which the web application is deployed.  
`cf create-service xsuaa application my-uaa -c security/xs-security.json`
2. Create an entry in the `manifest.yml` file, defining the service binding between the web application and the SAP Authorization and Trust Management service application plan service instance.  
`services:  
 -my-uaa`
3. Redeploy the web application.  
`cf push`

### Subscribe to the SaaS application

1. Log in to your Global Account.
2. Choose your subaccount.
3. Choose **Subscriptions**.
4. Choose the SaaS application that you want to subscribe to.
5. Choose **Subscribe**.
6. Retry accessing the SaaS application (it might be necessary to clear your cache).

## 7.1.5.3.8 Login Issues

### Symptom

You log on to SAP BTP via the Cloud Foundry CLI or the browser and you get an HTTP 401 error or see messages on the screen that indicate that your login attempt has failed.

### Reason and Prerequisites

Authentication in Cloud Foundry runs through the UAA service which tunnels the credentials supplied (such as username and password) towards a trusted Open ID Connect Identity Provider (IdP). SAP's default IdP

is the SAP ID service, which is available at <https://accounts.sap.com> for SAP ID Service, or at <https://account.sap.com> for SAP Universal ID.

To narrow down the origin of the issue, you need to check the individual components involved in the login process.

## Solution

### Identify whether a second factor (a one-time token) is mandatory for this user/landscape

1. Open the UAA service in the browser directly, depending on your region: For example, for the European region, go to <https://uaa.cf.eu10.hana.ondemand.com/> (For an overview of all Cloud Foundry landscape URLs, see [Regions](#)).  
If you are redirected to <https://accounts.sap.com> asking for two-factor authentication and a passcode, a second factor in the form of a one-time TOTP token is mandatory.
2. See [Activate TOTP Two-Factor Authentication](#) for information about how to configure a device to generate TOTP tokens; for example, with an authenticator app.
3. Log on to your Cloud Foundry environment using two-factor authentication.
  1. Type `cf login`.
  2. Enter your e-mail address.
  3. Enter your password in the following format: <your global password><one-time-token>

### 7.1.5.3.9 Cannot Add Role Templates to Predefined Role Collections

#### Symptom

You want to update your SAP Authorization and Trust Management service (XSUAA) instance and add role templates to a previously defined role collection in the `xs-security.json`. The service instance update fails with the error message:

```
Cannot add or remove role-template-references from role collection if already in use (app subscribed or role collection assigned to user). Please create a new role-collection with a different name instead.
```

#### Reason and Prerequisites

You tried to add a role template to an existing role collection that is already in use. If your application was either subscribed (for a multi-tenant enabled application) or the role collection in question was already assigned to a user, an update of the role collection is not possible anymore. The security reason for this is that role collections might have already been audited by subscribers and/or the security administrator assigned these role collections to users based on the initial state of the role collections. Changes to the role collections at a later point in time would render these audits invalid.

In general, you should never modify any authorization object (role template, role) that is already in use in a way that it grants additional authorization without explicit approval of the user and role administrator.

## Solution

### Create a new role collection with a different name

You can add an additional role collection with a different name to your `xs-security.json`. Administrators can then audit the new role collection and migrate the users to the updated role collection. To do this, follow these steps:

1. Add a new role collection with a different name to the `xs-security.json`.
2. Add/modify the role templates within that role collection according to your needs.
3. Update the SAP Authorization and Trust Management service service instance. See [Update Service Instances Using the SAP BTP Cockpit or Cloud Foundry Command Line Interface](#).
4. Assign the new role collection to the application users.
5. (Optional) When all users have been migrated to the new role collection, remove the old role collection from the `xs-security.json`.
6. (Optional) Update the SAP Authorization and Trust Management service service instance.

For more information about the creation of role collections and the `xs-security.json`, see [Create Role Collections \(with Predefined Roles\)](#) and [Application Security Descriptor Configuration Syntax](#).

## 7.1.5.3.10 409 Error Code on Deleting a Custom Identity Provider

### Symptom

- On deleting a custom identity provider (origin: `sap.custom`) in the SAP BTP cockpit, you get an error.
- If you open the developer console (F12) and try again, you will see a more detailed error message.

### Reason and Prerequisites

- This issue occurs when there are applications still present in the Identity Authentication service (IAS). For example, this can happen if you have created an identity service instance.

## Solution

1. Check the IAS tenant for which you want to delete the trust configuration in the SAP BTP cockpit.
2. You need to delete all the applications of the error message.

**Note:** Delete only those applications that you are sure are not needed anymore.

See [SAP Note 3148626](#) for more information.

### 7.1.5.3.11 Unexpected AuthnResponse : Existing authentication - <User>

## Symptom

After logging on to an application with SAML, the user presses the back button of the browser. This leads to the *Unexpected AuthnResponse : Existing authentication - <User>* error.

## Reason and Prerequisites

Using the back button leads to a second authentication request to the identity provider, with the same ID used for the initial logon. As a result, the identity provider issues a second authentication response for the same ID, which is then rejected.

## Solution

The error message displayed is the correct outcome of this action. If you want to trigger a logout by pressing the back button, this needs to be handled on the application side.

## **7.1.5.3.12 AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination>**

### **Symptom**

When a SAML session is dropped, the audit log shows AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination>. This particular error message can be found only in the audit log.

### **Reason and Prerequisites**

This error occurs when the identity provider takes more than 15 minutes to issue the authentication response or when the user waits with the logon action for more than 15 minutes on the identity provider side. After this time interval, the session information is dropped for performance and security reasons; and thus, the request and response do not match anymore.

### **Solution**

#### **Don't delay the logon action**

Ensure that the user login is within the timeframe of 15 minutes.

#### **Check the identity provider**

The identity provider may be out of sync; or if it consistently takes a long time to issue the authentication response, reduce this time and check.

## **7.1.5.3.13 InResponseToField of Response Doesn't correspond to the Sent Message**

### **Symptom**

When the user tries to log on with SAML or by refreshing the session, they see the *InResponseToField of Response doesn't correspond to the sent message* error.

## Reason and Prerequisites

This error occurs when handling an authentication response without a known corresponding request; for example, as the consequence of a session timeout. The timeout is a safeguard for performance and security reasons.

## Solution

### Don't delay the logon action

Ensure that the user login is within the timeframe of 15 minutes.

### Check the identity provider

The identity provider may be out of sync; or if it consistently takes a long time to issue the authentication response, reduce this time and check.

### Check the audit log

Check the audit log for the *AuthnRequest expired - ID: <RequestId> Destination: <IdPDestination>* error. If you find this error message, refer to [this Guided Answer](#).

For more information about the audit log, refer to [Auditing and Logging Information for SAP Authorization and Trust Management Service - SAP Help Portal](#).

## 7.1.5.3.14 Response Issue Time is Either too Old or with Date in the Future. Sync IdP to Match Skew <skew>

## Symptom

SAML logon doesn't work and the *Response issue time is either too old or with date in the future. Sync IdP to match skew <skew>* error message is displayed.

## Reason and Prerequisites

This error occurs when the time skew between the SAP Authorization and Trust Management service and the identity provider is larger than 60 seconds in either direction, or when after being issued, the authentication response takes more than 60 seconds to reach the SAP Authorization and Trust Management service.

## Solution

Check the time skew between the identity provider and the SAP Authorization and Trust Management service, and then sync the clock of the identity provider.

### 7.1.5.3.15 Trust Establishment Issues

#### Symptom

You could have trust establishment issues if you encounter one of the following symptoms:

- In the BTP Cloud Cockpit:
  - After clicking **Establish Trust**, an Identity Authentication (IAS) tenant is not displayed.
  - Failure after the IAS tenant is selected and confirmed by clicking **Establish Trust**.
  - **Establish Trust** button is not accessible (displayed in the inactive state).
  - Trust deletion fails.
- In the REST API:
  - Usage of the [REST API](#) (/sap/rest/identity-providers) returns status code 400, 401, or 404.
  - Trust deletion fails.

#### Reason and Prerequisites

To establish trust in an automated manner, the customer ID of the IAS tenant has to match the customer ID of the global account.

Consequently, the issues in trust establishment may be because of the following:

- The customer ID maintained in the IAS tenant does not match what is in the global account.
- Trust has already been established (IdP trust with origin `sap.custom` exists.).

## Solution

### Check that the IAS tenant is displayed

If an IAS tenant is not displayed, verify the following:

- The customer ID maintained in the IAS tenant matches what is in the customer ID of the global account.
  - IAS admin UI: *Applications & Resources > Tenant Settings > Customer ID*
  - Global account: Verify the value in the control center.

In case you require a new IAS tenant, see the corresponding [documentation](#). For more information, see [this blog article](#).

### Check that the "Establish Trust" button is accessible

If the Establish Trust button is grayed out, there is probably already an Identity Provider (IdP) trust with origin, sap.custom. To change the trust, you can delete the existing configuration and create a new one. Note that the trust deletion in the Cloud Cockpit implies the deletion of all existing shadow users (for this IdP).

If the Establish Trust button is not visible at all, it could be an issue with the Identity Authentication service (IAS) registry or the certificates. This is usually a temporary issue and you should retry accessing it later.

### Check the usage of the REST API

Depending on the status code, verify the following points:

- 400: Check the section, *Specific IAS tenant is not shown*. Further, verify that the /sap/rest/identity-providers/ias REST API returns the used IAS tenant.
- 401: REST API requires a token of the apiaccess service plan from your subaccount.
- 404: For newly created IAS tenants, it can take up to 2 hours until the DNS entry is propagated to all the DNS caches. Until that is done, you can get an error when trying to call the new IAS tenant. In such a case, wait and retry after 2 hours.

### Check if trust has already been established

- Verify that you are deleting a custom IdP (the default IdP cannot be deleted).
- Ensure that if you use the identity broker (or a SaaS application that uses it), it does not contain any application references or other applications.

## 7.1.5.3.16 Token Retrieval Fails With Status Code 401

### Symptom

You want to retrieve an access token from the Authorization and Trust Management service, but the request fails because of one of the following:

- Client authentication failed
- Bad credentials
- Client with requested ID does not exist
- The origin provided in *login\_hint* does not match an active Identity Provider (IdP) that supports resource-owner password grant flow.

## Reason and Prerequisites

The client or user authentication was not successful. This could happen if the client does not exist or if the client or user credentials are invalid.

- In the case of client authentication failing, during the authentication, the caller provides the `client_id` and the `client_secret`. If these are not correct, an error occurs. This is valid for both technical (`client_credentials`) and user authentication (e.g. `authorization_code`).
- Bad credentials could be the case in user flows and indicates that the user credentials (username/password) are incorrect.
- If a client with the requested ID does not exist, it signifies that the source of the `clientid` value should be checked.

You have to determine whether the issue is related to client or user authentication.

- For user authentication, verify whether token retrieval works for the `client_credentials` grant type.
- If not, the issue is related to client authentication.

## Solution

### User authentication

- Verify that the user and password combination is correct.
  - Open a browser and log on to the IdP (e.g. <https://accounts.sap.com>).
  - If the login fails, you need to use another user/password combination or you could change your password.
- In case of a Universal ID user:
  - Check [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface](#) and especially [SAP Note 3085908](#). You can define a separate password for the SAP ID service user that is linked to the Universal ID user, which you need to use for all the logins that involve the SAP ID service and where you have to provide a user name and password and cannot run the (redirected) login flow via the web browser.

### Client authentication

Client authentication

### Secrets

- Verify that the used secret is valid and that the corresponding binding/service key still exists.
- Depending on the credential type of the used binding/service key, do one of the following:
  - `instance-secret`: Ensure that the service instance of the client allows `instance-secret` in its security descriptor (within `credential-types`).
  - `binding-secret`: Ensure that the service instance of the client allows `binding-secret` in its security descriptor (within `credential-types`).
- If the application to which the service instance belongs was subscribed to from another subaccount, ensure that the application or service is onboarded correctly via the SAP SaaS Provisioning service.
- If the secret contains the + sign:

- Ensure that the secret is passed via POST body (and not in the Authorization header).
- If it's not possible to pass the secret via POST body, ensure that the username and password are correctly URL encoded before being passed via the header. Also, you need to set the x-CF-ENCODED-CREDENTIALS custom header to true to indicate that the Authorization and Trust Management service supports URL-encoded credentials.
- If you've recently updated to Spring version 5.6 or newer and use a Spring client for the client credential flow:
  - Set the x-CF-ENCODED-CREDENTIALS custom header to true.
  - Customize the Spring client such that it does not URL-encode credentials.

## X.509

- Verify that the certificate has been passed for authentication and that it is the correct one.
- Ensure that the service instance of the client allows x509 in its security descriptor (within credential-types).
- If the application to which the service instance belongs was subscribed to from another subaccount, ensure that the application or service is onboarded correctly via the SAP SaaS Provisioning service.
- For issues with the SSL handshake, create an incident in the [SAP Support Portal](#).

### No IdP that supports resource-owner password grant flow

- Resource-owner password grant flow is only supported with OpenID Connect (OIDC) IdPs, not SAML.
- Make sure to use the sap.default or sap.custom origin key.
- Origin keys can be checked in the Cockpit, under **Trust Configuration**.

## 7.1.5.3.17 Cockpit Displays HTTP Status 500 Error on Logon with Custom IdP User

### Symptom

### Reason and Prerequisites

After successfully logging on with a custom identity provider (IdP) user, the cockpit displays an error page: *HTTP Status 500 – Internal Server Error*.

The following are the known reasons for this error:

- The mail claim in the OpenID Connect (OIDC) token from the Identity Authentication (IAS) tenant, is empty. This could happen in the following cases:
  - The corporate identity provider (for example, MS Azure AD) is connected to the IAS and the user in the corporate IdP does not have an email address assigned.
  - The [Identity Authentication user store](#) is enabled, and the corporate identity provider is configured to send the email address of its user in another SAML attribute or OIDC claim than what is defined in the default attributes of the application in the Identity Authentication tenant.

- The Identity Authentication user store is disabled. This requires the corporate identity provider to send the user's email address in a SAML attribute or OIDC claim mail.
- The *mail*, *first\_name*, or *last\_name* claim in the OIDC token from IAS contains multiple values. This could happen in the following cases.
  - The IAS tenant or application is configured to use a corporate IdP with identity federation enabled and the user also exists in the IAS user store, or the corporate IdP sends multiple values in the first place.
  - In the case of the *mail* claim, the corporate IdPs OIDC token or SAML assertion does not send the email address in all lowercase. IAS always converts the email address to lowercase for local users.
- The issuer in the connected IAS tenant was changed after the trust to that tenant was established.

## Solution

### Configure the corporate identity provider

As typically, either some attribute is missing or has multiple values instead of a single one, configure the corporate identity provider to correctly send the mail claim or required attributes.

- If a platform user has problems when logging on with the a custom identity provider, refer to the [IAS Troubleshooting Logs \(OIDC\)](#) for that affected user.
- In particular, check if there is exactly one value (and the correct value) for the [specified attributes](#).

### Check the issuer in the IAS

The issuer in the IAS must not be changed after the trust has been established. This is also documented as a restriction [here](#).

## 7.1.5.3.18 IAS Application Reference isn't Created in Your IAS Tenant

### Symptom

When trying to log on to the global account with a custom identity provider (IdP) user, the Identity Authentication (IAS) tenant responds with the following error message: *OpenID provider cannot process the request because the configuration is incorrect. Please contact your system administrator.*

### Reason and Prerequisites

Most likely, you configured a platform IdP earlier and deleted the resulting IAS application reference (SAP Business Technology Platform, *btp-cockpit*) from the IAS tenant.

## Solution

### Delete and recreate the trust to the IAS tenant

In your global account, delete and recreate the trust to the IAS tenant.

## 7.1.5.3.19 User Base Doesn't Appear in Existing Neo Subaccounts

### Symptom

When trying to add new members, Neo member management doesn't have the custom identity provider configured in the global account, as the user base.

### Reason and Prerequisites

For subaccounts in the Neo environment, the identity provider will be offered in the value help only if a user in that identity provider has created at least one Neo subaccount in the corresponding global account and Neo region. For more information about this, see [here](#).

## Solution

### Create at least one new Neo subaccount

A user from the configured custom identity provider needs to create at least one new Neo subaccount in the Neo region where the user base selection is missing.

## 7.1.5.3.20 User from Corporate IdP Cannot Log on to Neo Subaccount

### Symptom

When a user with a corporate identity provider (IdP) logs on to the cockpit and tries to navigate to a Neo subaccount, the Identity Authentication (IAS) tenant shows the logon screen with the *E-Mail or User Name* and

Password fields. However, with a properly connected and configured corporate IdP, the IAS logon screen should show only the *E-Mail or User Name* field.

## Reason and Prerequisites

For all the Neo datacenters where you have subaccounts, you might have to configure a separate SAML application (for example, SAP Cloud Platform *hhw9g3jrx*) in the IAS tenant. This behavior is also documented [here](#).

- A custom IdP with a corporate identity provider (e.g. MS Azure AD) is configured in the global account.
- The affected users from the corporate IdP can log on to the cockpit (e.g. <https://cockpit.eu10.hana.ondemand.com/cockpit/?idp=ar9ibaxhm.accounts.ondemand.com>).
- The affected users are configured as members of the Neo subaccount, with the correct user base.

## Solution

Find the correct SAML application in the configured IAS tenant and configure conditional authentication accordingly. Each Neo region where custom IdP is configured has a separate SAML application in IAS that needs to be configured.

### 7.1.5.3.21 Share Authorization and Trust Management service instances

## Symptom

You want to share Authorization and Trust Management service instances among different Cloud Foundry spaces or environments.

## Reason and Prerequisites

- You use an Authorization and Trust Management service instance of the *application* type.
- You want to use the service instance in multiple spaces or environments, without recreating it multiple times.

## Solution

Use the SAP BTP command line interface (btp CLI) to share the *application* plan.

For more information, see [Sharing and Unsharing Service Instances](#).

### 7.1.5.3.22 Invalid "redirect\_uri"

## Symptom

After logging on, you are redirected to a *Where-To?* page with one of the following error messages:

- The *redirect\_uri* is invalid
- The *redirect\_uri* has an invalid domain
- Invalid redirect: {*URI*} does not match one of the registered values
- Client registration is missing *redirect\_uri*
- Client registration contains *invalid redirect\_uri*

## Reason and Prerequisites

The **redirect\_uris** configuration value for the used service instance has not been properly configured.

## Solution

1. Identify the affected application. The affected application is the application on which you want to be redirected after the logon process.
2. Identify the affected service instance of the xsuaa type.
  - Cloud Cockpit: Find out which service instance of the xsuaa type was bound to the application.
  - CF CLI: Check the environment of the application, and then find xsuaa to retrieve the name of the service instance: `cf env <application-name>`
3. Identify the xsappname of the service instance (which is mandatory for *xs-security.json*).
  - Existing *xs-security.json*: You can find the xsappname of the *xs-security.json* file that was used to create the service instance.
  - Cloud Cockpit: Create a service key and retrieve the xsappname from it; remove the section after the exclamation mark - for example, *myapp!b123* -> xsappname: *myapp*. For more information, see [SAP Note 3319310](#).
  - CF CLI: Check the environment of the application (see step 2), retrieve the xsappname from the corresponding service, and then remove the section after the exclamation mark.

4. Prepare the security descriptor (`xs-security.json`).
  - For more information, see [Application Security Descriptor Configuration Syntax](#).
5. Update the `redirect_uris` attribute of the corresponding service instance (within `oauth2-configuration`).
  - Cloud Cockpit: Click **Update** in the service, and then provide the `xs-security.json` file, as required.
  - CF CLI: Use `cf update-service` to update the service instance and provide the required `xs-security.json` file: `cf update-service <instance-name> -c <path-to-xs-security.json>`

**Note:** It is important to provide the complete `xs-security.json` file during the update as partial updates are not supported.

### 7.1.5.3.23 400 Error: Call to /oauth/token Was Not Successful

#### Symptom

You tried to log on to your application and got an error related to the token size exceeding some limits (for example, the HTTP header size).

#### Reason and Prerequisites

By default, access and refresh tokens include SAML groups and role collections. If you have a lot of SAML groups or role collections, the token size increases, and this can cause the JWT to exceed the size limit of 16 K. This results in a 400 error and the user not being able to log in.

#### Solution

##### Keep the JWT size to a minimum

Application security is maintained in the application security descriptor file (`xs-security.json`). In the syntax of the application security descriptor file, do the following to keep the JWT size to a minimum.

- Set the value of the `system-attributes` parameter to an empty array, like this: `"system-attributes": []`
- Configure only the required groups in the Identity Provider (IdP).

For more information regarding the `xs-security.json` file, see [Application Security Descriptor Configuration Syntax](#).

## 7.1.6 Security Considerations for the SAP Authorization and Trust Management Service

Decisions you make when using or administrating the SAP Authorization and Trust Management service (XSUAA) can have an impact on the security of your applications. The information provided is meant to help you decide.

### Related Information

[Configuration Options for the SAP Authorization and Trust Management Service \[page 3078\]](#)

### 7.1.6.1 Implications of Using iFrames

By default, applications of the subaccount, including login pages of the SAP Authorization and Trust Management service can't be framed by other applications for security reasons.

As the security administrator of a subaccount, you can allow the embedding applications of the subaccount in an iFrame.

To configure the trusted domains, use one of the following methods:

- Enter trusted domains in the SAP BTP cockpit.  
For more information, see [Configure Trusted Domains for Multi-environment Subaccounts \[page 2315\]](#).
- Access the Security Settings API.

```
"iframeDomains": "https://store.example.com"
```

For more information, see [Security Settings API](#) on [SAP Business Accelerator Hub](#).

#### → Remember

To use iFrames with SAP Authorization and Trust Management service, configure the embedding of the login page of your identity provider. Not all identity providers support framing. This function only works if the identity provider supports framing and is configured, too.

To use iFrames with other applications of the subaccount, which support this feature, configure the domains used by these applications.

To configure iFrames for SAP Cloud Identity Services, see [Configure Trusted Domains](#) in the documentation of SAP Cloud Identity Services.

When configured, the SAP Authorization and Trust Management service sends a content security policy header allowing framing of the application for the domains you specified.

For more information about content security policies, see the World Wide Web Consortium (W3C) recommendation [Content Security Policy](#).

This configuration has several security implications like:

- Clickjacking attacks on the application as the URL of the application isn't visible.
- JavaScript-based cross-site scripting vulnerabilities of older browser versions.
- Third-party cookies are required when using different domains for authentication with the SAP Authorization and Trust Management service, the identity provider, and the application.  
See also [Third-Party Cookies and SAP Authorization and Trust Management Service \[page 3080\]](#).

Ensure that no attacker can add malicious web pages or Javascript to any of the hosts allowed to frame the applications. This recommendation also includes hosts that act as reverse proxies, where an attacker can put their content on a different host behind the reverse proxy. Ensure that everything exposed by those framing hosts is safe.

## 7.1.6.2 Listing Allowed Redirect URIs

When developing your application, provide the list of the redirect URIs that the application needs when redirecting, for example during login or logout. Enter the list in the `redirect-uris` property of the application security descriptor (`xs-security.json`).

At runtime, the SAP Authorization and Trust Management service checks whether the redirect URI is listed in the property and rejects any other URIs. If your redirect isn't on this list, including wildcards, the service won't redirect users there.

The following is an example of a configuration of the `redirect-uris` property in an `xs-security.json`.

```
"oauth2-configuration": {  
    ...  
    "redirect-uris": [  
        "https://myapplication.cfapps.eu10-004.hana.ondemand.com",  
        "https://myapplication.cfapps.eu10-004.hana.ondemand.com/my/content",  
        "https://myapplication.mydomain.com",  
        "https://myapplication.mydomain.com/my/content"  
    ],  
    ...  
}
```

### → Recommendation

Set the `redirect-uris` property to restrict access as much as possible.

We support explicit wildcards, namely domain relaxing and arbitrary paths. For example: "`https://*.mydomain.com/callback/**`"

### ⚠ Caution

If you use wildcards, we recommend that you make your URIs as specific as possible. By using wildcards, you open up the redirect for multiple websites. Wildcards increase the risk of redirecting to malicious websites.

### ⓘ Note

In cloud landscapes, only `localhost` is always allowed by default as a redirect URI.

## Related Information

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

[Configure Redirect URLs for Browser Logout \[page 535\]](#)

### 7.1.6.3 Rotating Secrets

By default, all bindings of a service instance of the SAP Authorization and Trust Management service share a single instance secret. (Exception: Service instances with the apiaccess plan use a secret per binding by default.) In the application security descriptor (`xs-security.json`), enable individual secrets for each binding of a service instance.

We recommend this configuration so that you can rotate the secret of a binding without affecting the other bindings of the service instance. We also recommend that you rotate secrets regularly.

For more information, see [Managing Secrets of the SAP Authorization and Trust Management Service \[page 2294\]](#).

```
"oauth2-configuration": {  
    ...  
    "credential-types": [ "binding-secret" ],  
    ...  
}
```

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

### 7.1.6.4 Setting Token Policy

By default, SAP Authorization and Trust Management service sets the validity of tokens as shown in the following table.

Default Validity of Tokens

Token Type	Validity
Access tokens	Default: 43200 seconds (12 hours)
Refresh tokens	Default: 604800 seconds (7 days)

On request, SAP Authorization and Trust Management service issues access and refresh tokens.

With a valid access token, you can access a protected resource. Once an access token expires, you can get new access tokens with a refresh token. Once the refresh token expires, you must reauthenticate and request new access and refresh tokens.

See also [Authorization](#) for information on how tokens affect authorizations.

## → Recommendation

Relaxing the token policy means that users reauthenticate less. However, increasing the token validity also means that if a malicious user manages to steal a token, that malicious user has access until the token expires. Keep token validity as short as possible, but not less than 30 minutes.

To change token validity, use one of the following methods:

- To target a specific instance of the service, change the values of the *token-validity* or *refresh-token-validity* parameters in the application security descriptor (`xs-security.json`).

```
"oauth2-configuration": {  
    ...  
    "token-validity": 1800,  
    "refresh-token-validity": 43200,  
    ...  
}
```

### ⓘ Note

This change applies to applications, which consume or subscribe to this instance. These values override the values set for the subaccount.

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

- To target the subaccount in general, access the Security Settings API.

```
"tokenPolicySettings": {  
    ...  
    "accessTokenValidity": 1800,  
    "refreshTokenValidity": 43200,  
    ...  
}
```

### ⓘ Note

This change applies to all service instances in the subaccount that haven't set a specific value in the application security descriptor (`xs-security.json`).

For more information, see [Security Settings API](#) on [SAP Business Accelerator Hub](#).

## 7.1.6.5 Access to REST APIs

The `apiaccess` plan provides administrator-level access to REST APIs of the SAP Authorization and Trust Management service. Ensure that only administrators have administrator access to Cloud Foundry spaces where service instances with this plan exist.

To use this service plan, create a separate Cloud Foundry space in your subaccount. Only assign administrator users the *Space Manager* and *Space Developer* roles in this space as well as the *Org Manager* role in the parent Cloud Foundry org. Use this Cloud Foundry space to create the `apiaccess` plan service instance.

Administrators are any SAP BTP cockpit users with the *User and Role Administrator* role. This role is part of the *Subaccount Administrator* role collection, but can be included in other role collections.

## Related Information

[About Roles in the Cloud Foundry Environment \[page 2431\]](#)

[Default Role Collections of SAP BTP \[page 3090\]](#)

### 7.1.6.6 Training Business Users How to Handle Session Timeouts

Administrators should make business users aware of a possible security risk after single logoff. A business user might get a message saying that this user is being logged off from the application although the identity provider session was not terminated.

The reason for this message comes from a timeout mismatch between the application and the SAP Authorization and Trust Management service. The timeout of the SAP Authorization and Trust Management service is currently 30 minutes. After the timeout, the SAP Authorization and Trust Management service does not forward the logoff request to the identity provider. This situation incurs a security risk.

#### → Recommendation

To remedy this security risk, advise the business users to access the application URL again, log on to the application, and log off right away, or log off directly from the identity provider.

### 7.1.7 Configuration Options for the SAP Authorization and Trust Management Service

The following configuration options enable you to manipulate the operation of the SAP Authorization and Trust Management service (XSUAA). Set these options in the application security descriptor (`xs-security.json`) at design time for your application.

## Related Information

[Security Considerations for the SAP Authorization and Trust Management Service \[page 3074\]](#)

## 7.1.7.1 Disabling System Attributes

Access tokens can become overloaded with too many SAML group or role collection attributes. By default, the SAP Authorization and Trust Management service includes all system attributes in the tokens the service issues.

If the number of attributes cause the token size to exceed 16k, you get HTTP error code 400. To avoid this problem, set the [system-attributes](#) parameter in the application security descriptor (`xs-security.json`).

```
"oauth2-configuration": {  
    ...  
    "system-attributes": [],  
    ...  
}
```

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

See also [400 Error: Call to /oauth/token Was Not Successful](#) in *Guided Answers*.

## 7.1.7.2 Preconfigured Role Collections

In the application security descriptor (`xs-security.json`) of your application, you define the authorizations needed to access your application. You define these authorizations as scopes within roles and role templates. From these entities, the administrator can build role collections and assign the role collections to users.

We recommend that you also define role collections for administrators to consume with your application. Delivering role collections with your application has the following advantages:

- Time savings for the administrator.  
The administrator of the subaccount can consume your application more quickly.
- Offer examples of well-defined role collections.  
The administrator can learn from your example, the security model of your application and design his or her own role collections based upon that model.

For more information about authorizations, see [Authorization Entities \[page 3036\]](#).

```
"role-collections": [  
    {  
        "name": "Employee",  
        "description": "Employee roles",  
        "role-template-references": [  
            "$XSAPPNAME.Employee"  
        ]  
    }  
]
```

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

### 7.1.7.3 Asynchronous Processing for Multi-Tenant Applications

After your application has more than 50 subscriptions, the time required to update instances of SAP Authorization and Trust Management service bound to your application increases. This delay can run into the timeouts set by the Cloud Controller of Cloud Foundry and SAP Service Manager.

To avoid timeout, enable asynchronous processing in your application security descriptor (`xs-security.json`). To enable asynchronous processing, set the `xsenableasyncservice` parameter to `true`.

```
"xsenableasyncservice": "true"
```

For more information about the application security descriptor, see [Application Security Descriptor Configuration Syntax \[page 545\]](#).

### 7.1.8 Third-Party Cookies and SAP Authorization and Trust Management Service

Browser manufacturers are phasing out support for third-party cookies. This change can prevent embedded BTP applications and their logon screens from functioning.

To prevent your logon scenarios from failing, make sure that all multi-environment subaccounts, which contain these embedded applications, are configured to use a tenant of SAP Cloud Identity Services as the trusted identity provider.

#### ⚠ Caution

Migrate your trust before the browser vendors phase out support.

Trusting your SAP Cloud Identity Services tenant enables your applications to integrate with the overall solution provided by SAP.

For more information about the overall solution, see [SAP BTP and Third-Party Cookies Deprecation](#) and SAP Note [3409306](#).

To migrate an existing trust configuration, see [Migration from SAML Trust to OpenID Connect Trust with Identity Authentication](#).

If your framing and framed applications run under different domains, you must still maintain trust between the domains as before. For more information, see [Configure Trusted Domains for SAP Authorization and Trust Management Service \[Feature Set B\]](#).

#### → Recommendation

Test your solutions before third-party cookies have been phased out. To test how your solutions function once third-party cookies have been phased out, enable the `partitionedCookies` property of the Security Settings API of the SAP Authorization and Trust Management service and run your tests. If your business scenarios fail, set `partitionedCookies` to false to help troubleshoot the source of the problem.

For more information, see the following:

- How to test: [Preparing and Testing Your Solution for Third-Party Cookie Phase-Out](#)
- How to use the API: [Accessing Administration Using APIs of the SAP Authorization and Trust Management Service](#)

In subaccounts created before May 30, 2024, the `partitionedCookies` property has a default value of `false`. In subaccounts created afterward, the property has a default value of `true`. When Google begins large-scale phase-out of third-party cookies, we plan to set this property to `true` and read-only in all subaccounts.

## Related Information

[Trust and Federation with Identity Providers \[page 2204\]](#)

## 7.1.9 Security Recommendations for SAP Authorization and Trust Management Service

This service contributes to a central list of security recommendations for SAP BTP.

See the [list of security recommendations](#) for the SAP Authorization and Trust Management service.

The following sections present an overview of these recommendations according to your role.

### Account Administrators

Account administrators manage access to SAP BTP for their organization. They also manage the apps to which their organization is subscribed.

Security Recommendations for Account Administrators

Summary	Recommendation
Use custom identity providers for better integration and policy enforcement.	<a href="#">BTP-UAA-0008</a>
Check your custom identity providers.	<a href="#">BTP-UAA-0015</a>
Synchronize the user data with a provisioning service.	<a href="#">BTP-UAA-0016</a>
Check which users have critical roles.	<a href="#">BTP-UAA-0017</a>
Check that only administrators have access to REST APIs.	<a href="#">BTP-UAA-0022</a>

Summary	Recommendation
Remove users who haven't logged on for a long time.	<a href="#">BTP-UAA-0010</a>
Implement a data deletion strategy.	<a href="#">BTP-UAA-0011</a>
Check the validity configuration for access tokens.	<a href="#">BTP-UAA-0005</a>
Rotate the signing keys for access tokens.	<a href="#">BTP-UAA-0019</a>
Rotate the signing keys for the SAML protocol.	<a href="#">BTP-UAA-0020</a>
Ensure understandable links for identity providers on logon pages.	<a href="#">BTP-UAA-0018</a>
Archive audit log entries.	<a href="#">BTP-UAA-0012</a>
Integrate the audit log with your corporate event management system.	<a href="#">BTP-UAA-0013</a>
If you embed the login page of this service, check where you allow the page to be embedded.	<a href="#">BTP-UAA-0001</a>
Train business users how to handle session timeouts	<a href="#">BTP-UAA-0023</a>

## Developers

Developers create new applications for their organization.

Security Recommendations for Developers

Summary	Recommendation
Enable individual secrets for each binding of a service instance.	<a href="#">BTP-UAA-0003</a>
Set the <code>redirect-uris</code> parameter to restrict access as much as possible.	<a href="#">BTP-UAA-0002</a>
Check your custom code for updates to client libraries offered by SAP in your dependencies.	<a href="#">BTP-UAA-0021</a>

## Platform-as-a-Service Operators

Platform-as-a-service operators manage applications developed by their organization.

Summary	Recommendation
Automate your deployment process to cope with the lifetime of the X.509 certificates.	<a href="#">BTP-UAA-0014</a>

## 7.1.10 Rate Limiting

This section provides information on the rate limiting in the SAP Authorization and Trust Management service.

The SAP Authorization and Trust Management service acts as an OAuth authorization server and issues access tokens with a long validity. These tokens are meant to be reused. However, some OAuth clients don't reuse them and thus cause a high load. The SAP Authorization and Trust Management service is scaled per landscape to handle the load.

To protect the SAP Authorization and Trust Management service from overload of misbehaving OAuth clients, rate limiting has been introduced. The limit is per subaccount.

- Requests sent at a lower rate are processed directly.
- When the limit for a subaccount is exceeded, the requests are queued and sent to the SAP Authorization and Trust Management service at a maximum rate (see table). The response times increase because of the queuing.
- If too many requests are queued so that the response time due to queuing exceeds a certain time, the service sends an HTTP 429 response code. This response also contains the `http Retry-After` header, which indicates when the client can retry.

### → Recommendation

There might be company events that trigger a lot of users to log on and try to access one special resource, such as the launch of a special employee discount sale for a certain day. Therefore, it's very likely that a high load of users will try to log on to the special discount website at the announced time. This means that you can foresee that users might run into HTTP 429 error response codes.

- We recommend sending the announcement emails in waves. By doing so, you can avoid a situation where almost all users try to log on at about the same time.
- Additionally, it's helpful to handle the HTTP 429 error message by also sending a clear text message like `We are sorry, but the server is currently experiencing high demand. Please try again later.` Such a message gives the users guidance of what to do.

### ⓘ Note

Since rate limiting is per subaccount, SaaS applications probably don't notice rate limiting because the load is distributed across different SaaS subaccounts.

### ⚠ Restriction

The following table lists sizing restrictions when using APIs.

Sizing Restrictions	
Element	Maximum Size
HTTP header	16 KB
POST body	1 MB

## Token Endpoint

The token endpoint enables you to authenticate with OAuth 2.0.

### Token Endpoint

Endpoint	Subaccount Limit	Effect
<code>https://&lt;subdomain&gt;.authentication.&lt;landscape&gt;/oauth/token</code>	Up to 60 requests per second	The requests are executed at once.
<code>https://&lt;subdomain&gt;.authentication.&lt;landscape&gt;/oauth/token</code>	Exceeding 60 requests per second	The requests are queued and then executed.
<code>https://&lt;subdomain&gt;.authentication.&lt;landscape&gt;/oauth/token</code>	Significantly exceeding 60 requests per second	An HTTP 429 Too Many Requests response status code is sent.
<code>https://&lt;subdomain&gt;.authentication.cert.&lt;landscape&gt;/oauth/token</code>	Up to 60 requests per second	The requests are executed at once.
<code>https://&lt;subdomain&gt;.authentication.cert.&lt;landscape&gt;/oauth/token</code>	Exceeding 60 requests per second	The requests are queued and then executed.
<code>https://&lt;subdomain&gt;.authentication.cert.&lt;landscape&gt;/oauth/token</code>	Significantly exceeding 60 requests per second	An HTTP 429 Too Many Requests response status code is sent.

## Authorization, Security Settings, and Identity Provider Management APIs

The authorization, security settings, and identity provider management APIs enable you to manage service instances, roles, role templates, role collections, and identity providers. You find the APIs of SAP Authorization and Trust Management service and of the security settings in the [SAP Business Accelerator Hub](#).

Authorization, Security Settings, and Identity Provider APIs

Endpoint	Subaccount Limit	Effect
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/authorization/v2</code> (Including all APIs with subpaths, for example <code>sap/rest/authorization/v2/apps</code> )	Up to 45 requests per second	The requests are executed at once.
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/authorization/v2</code> (Including all APIs with subpaths, for example <code>sap/rest/authorization/v2/apps</code> )	Exceeding 45 requests per second	The requests are queued and then executed.
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/authorization/v2</code> (Including all APIs with subpaths, for example <code>sap/rest/authorization/v2/apps</code> )	Significantly exceeding 45 requests per second	An HTTP 429 Too Many Requests response status code is sent.
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/identity-providers</code> (Including all APIs with subpaths, for example <code>sap/rest/identity-providers/{id}</code> )	Up to 30 requests per second	The requests are executed at once.

Endpoint	Subaccount Limit	Effect
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/identity-providers</code>  (Including all APIs with subpaths, for example <code>sap/rest/identity-providers/{id}</code> )	Exceeding 30 requests per second	The requests are queued and then executed.
<code>https://api.authentication.&lt;landscape&gt;/sap/rest/identity-providers</code>  (Including all APIs with subpaths, for example <code>sap/rest/identity-providers/{id}</code> )	Significantly exceeding 30 requests per second	An HTTP 429 Too Many Requests response status code is sent.

## User Management (SCIM) APIs

The user management (SCIM) APIs enable you to manage shadow users and role collections. You find the APIs of SAP Authorization and Trust Management service in the [SAP Business Accelerator Hub](#).

### User Management (SCIM) APIs

Endpoint	Subaccount Limit	Effect
<code>https://api.authentication.&lt;landscape&gt;/groups</code>  (Including all APIs with subpaths, for example <code>sap/rest/groups/{id}</code> )	Up to 3 requests per second	The requests are executed at once.
<code>https://api.authentication.&lt;landscape&gt;/groups</code>  (Including all APIs with subpaths, for example <code>sap/rest/groups/{id}</code> )	Exceeding 3 requests per second	The requests are queued and then executed.

Endpoint	Subaccount Limit	Effect
<code>https://api.authentication.&lt;landscape&gt;/groups</code> (Including all APIs with subpaths, for example <code>sap/rest/groups/{id}</code> )	Significantly exceeding 3 requests per second	An HTTP 429 Too Many Requests response status code is sent.
<code>https://api.authentication.&lt;landscape&gt;/users</code> (Including all APIs with subpaths, for example <code>sap/rest/users/{id}</code> )	Up to 3 requests per second	The requests are executed at once.
<code>https://api.authentication.&lt;landscape&gt;/users</code> (Including all APIs with subpaths, for example <code>sap/rest/users/{id}</code> )	Exceeding 3 requests per second	The requests are queued and then executed.
<code>https://api.authentication.&lt;landscape&gt;/users</code> (Including all APIs with subpaths, for example <code>sap/rest/users/{id}</code> )	Significantly exceeding 3 requests per second	An HTTP 429 Too Many Requests response status code is sent.

## Related Information

[Application Security Descriptor Configuration Syntax \[page 545\]](#)

[Accessing Administration Using APIs of the SAP Authorization and Trust Management Service \[page 2416\]](#)

## 7.1.11 Limits for Technical Artifacts of the SAP Authorization and Trust Management Service

To improve the resiliency of the SAP Authorization and Trust Management service, we have introduced limitations on technical artifacts of the service.

### Limits for the Service Broker

We allow a maximum of service instances per subaccount.

There is also a maximum of service bindings and service keys in total per service instance. The service rejects attempts to add more bindings or service keys. For more information, see [Service Instance Secrets \[page 2294\]](#).

Limits for the Service Broker

Service Broker Resource	Upper Limit
Service instances per subaccount	2500
Service bindings and service keys in total per service instances	1000

### Limits for the Application Security Descriptor

There are limits that apply when you define properties in the application security descriptor.

You define properties in the `xs-security.json` application security descriptor file for creating or updating service instances. Take care that you don't define more properties than specified in the table.

Limits for Properties

Property	Upper Limit
attributes	50
role-collections	250
role-templates	500
scopes	500

Property	Upper Limit
scopes (granted)	250

**① Note**

Granted means that the scope object refers to applications that are specified using granted-apps or grant-as-authority-to-apps properties

## Limits for the Global Account

It is possible to configure up to 3 custom identity providers for platform users in a global account.

## Limits for the Subaccount

There are limits that apply to the subaccount. Administrators can use application roles and create their own roles. They can create and manage role collections, add roles to them, and assign the role collections to business users.

For more information, see [Building Roles and Role Collections for Applications \[page 2284\]](#) and [Working with Role Collections \[page 2274\]](#).

Limits for Security Artifacts

Security Artifact	Upper Limit
Roles created by administrators	2500

**① Note**

Roles that are automatically generated when a service instance was created or updated are not counted here.

Role collections created by administrators	5000
--	------

When establishing trust, subaccount administrators must observe the limits for identity providers.

Limits for Identity Providers

Identity Provider	Upper Limit
SAML identity providers	50
User attribute mappings defined in a subaccount	5000

## 7.1.12 Configuring Backup

For SAP Authorization and Trust Management Service, you can backup the service configuration, the authorizations, and the configuration of the identity provider using APIs.

For more information, see [SAP BTP Service Configuration Backups Managed by Customers](#).

## 7.1.13 Accessibility Features in SAP Authorization and Trust Management Service

To optimize your experience of the SAP Authorization and Trust Management service, SAP Business Technology Platform provides features and settings that help you use the software efficiently.

### ⓘ Note

SAP Authorization and Trust Management service runs on the SAP BTP cockpit. For this reason, accessibility features for SAP BTP cockpit apply. For more information, see the accessibility documentation for SAP BTP cockpit on SAP Help Portal in [Accessibility Features in SAP BTP cockpit \[page 2141\]](#).

For more information on screen reader support and keyboard shortcuts, see the [Accessibility for End Users](#).

## 7.2 Default Role Collections of SAP BTP

The following table displays the default role collections available with SAP BTP after initially deploying your accounts.

Default Role Collections Including Roles and Role Templates

Role Collection	Role Name	Role Template	App ID	Role Description
Global Account Administrator	Global Account Admin	GlobalAccount_Admin	cis-central!<suffix>	Role for global account members with read-write authorizations for core commercialization operations, such as updating global accounts, setting entitlements, and creating, updating, and deleting subaccounts.

<b>Role Collection</b>	<b>Role Name</b>	<b>Role Template</b>	<b>App ID</b>	<b>Role Description</b>
Global Account Administrator	Global Account Usage Reporting Viewer	GlobalAccount_Usage_Reportng_Viswer	uas!<suffix>	Role for global account members with read-only authorizations for core commercialization operations, such as viewing global account usage information.
Global Account Administrator	User and Role Administrator	xsuaa_admin	xsuaa!<suffix>	Manage authorizations, trusted identity providers, and users.
Global Account Administrator	System Landscape Administrator	GlobalAccount_System_Landscape_Administrator	cmp!<suffix>	Administrative access to systems and scenario-related resources.
Global Account Viewer	System Landscape Viewer	GlobalAccount_System_Landscape_Viswer	extension-service-cmp!<suffix>	Viewer access to systems and scenario-related resources.
Subaccount Administrator	Cloud Connector Administrator	Cloud_Connector_Administrator	connectivity!<suffix>	Operate the data transmission tunnels used by the Cloud connector.
Subaccount Administrator	Destination Administrator	Destination_Administrator	destination-xsapp-name!<suffix>	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Subaccount Administrator	Subaccount Admin	Subaccount_Admin	cis-local!<suffix>	Role for subaccount members with read-write authorizations for core commercialization operations, such as viewing subaccount entitlements, and creating and deleting environment instances.
Subaccount Administrator	User and Role Administrator	xsuaa_admin	xsuaa!<suffix>	Manage authorizations, trusted identity providers, and users.
Subaccount Administrator	Subaccount Service Administrator	Subaccount_Service_Administrator	service-manager!<suffix>	Administrative access to service brokers and environments on a subaccount level.

Role Collection	Role Name	Role Template	App ID	Role Description
Global Account Viewer	Global Account Viewer	GlobalAccount_Visitor	cis-central!<suffix>	Role for global account members with read-only authorizations for core commercialization operations, such as viewing global accounts, subaccounts, entitlements, and regions.
Global Account Viewer	Global Account Usage Reporting Viewer	GlobalAccount_Usage_ReportViewer	uas!<suffix>	Role for global account members with read-only authorizations for core commercialization operations, such as viewing global account usage information.
Global Account Viewer	User and Role Auditor	xsuaa_auditor	xsuaa!<suffix>	Read-only access for authorizations, trusted identity providers, and users.
Subaccount Viewer	Cloud Connector Auditor	Cloud_Connector_Auditor	connectivity!<suffix>	View the data transmission tunnels used by the Cloud connector to communicate with back-end systems.
Subaccount Viewer	Destination Viewer	Destination_Visitor	destination-xsapp-name!<suffix>	View destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Subaccount Viewer	Subaccount Viewer	Subaccount_Visitor	cis-local!<suffix>	Role for subaccount members with read-only authorizations for core commercialization operations, such as viewing subaccount entitlements, details of environment instances, and job results.
Subaccount Viewer	User and Role Auditor	xsuaa_auditor	xsuaa!<suffix>	Read-only access for authorizations, trusted identity providers, and users.

Role Collection	Role Name	Role Template	App ID	Role Description
Subaccount Viewer	Subaccount Service Auditor	Subaccount_Service_Auditor	service-manager!<suffix>	Read-only access to service brokers and environments on a subaccount level
Subaccount Service Administrator	Subaccount Service Administrator	Subaccount_Service_Administrator	service-manager!<suffix>	Administrative access to service brokers and environments on a subaccount level.
Cloud Connector Administrator	Cloud Connector Administrator	Cloud_Connector_Administrator	connectivity!<suffix>	Operate the data transmission tunnels used by the Cloud connector.
Destination Administrator	Destination Administrator	Destination_Administrator	destination-xsapp-name!<suffix>	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Connectivity and Destination Administrator	Cloud Connector Administrator	Cloud_Connector_Administrator	connectivity!<suffix>	Operate the data transmission tunnels used by the Cloud connector.
Connectivity and Destination Administrator	Destination Administrator	Destination_Administrator	destination-xsapp-name!<suffix>	Manage destination configurations, certificates and subaccount trust via the Destination editor in the SAP BTP cockpit.
Directory Administrator	Directory Admin	Directory_Admin	cis-central!<suffix>	Role for directory members with read-write authorizations for core commercialization operations, such as updating directories, setting entitlements, and creating, updating, and deleting subaccounts.
Directory Administrator	User and Role Administrator	xsuaa_admin	xsuaa!<suffix>	Manage authorizations, trusted identity providers, and users.

Role Collection	Role Name	Role Template	App ID	Role Description
Directory Administrator	Directory Usage Reporting Viewer	Directory_Usage_Reportng_Visitor	uas!<suffix>	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directory usage information.
Directory Viewer	Directory Viewer	Directory_Visitor	cis-central!<suffix>	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directories, subaccounts, entitlements, and regions.
Directory Viewer	User and Role Auditor	xsuaa_auditor	xsuaa!<suffix>	Read-only access for authorizations, trusted identity providers, and users.
Directory Viewer	Directory Usage Reporting Viewer	Directory_Usage_Reportng_Visitor	uas!<suffix>	Role for directory members with read-only authorizations for core commercialization operations, such as viewing directory usage information.

## 7.3 Trusted Certificate Authentication

All components of SAP BTP rely on a list of trusted Certificate Authorities (CAs), which differ depending on the runtimes used.

For your scenarios with SAP BTP, be aware of this list of CAs to ensure that you can provide trusted certificates that are signed by a trusted CA.

A list of all root CAs approved by SAP Global Security is available in SAP Note [2801396](#) (SAP Global Trust List).

## 7.4 Principal Propagation

Exchange user ID information between systems or environments in SAP BTP.

### In This Section

- [Principal Propagation from the Cloud Foundry to the Neo Environment \[page 3102\]](#)
- [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 3095\]](#)

### Other Principal Propagation Scenarios

- [On-Premise User Store](#)
- [Principal Propagation to OAuth-Protected Applications](#)
- [Connectivity in the Cloud Foundry Environment: Principal Propagation](#)
- [Connectivity in the Neo Environment: Principal Propagation](#)

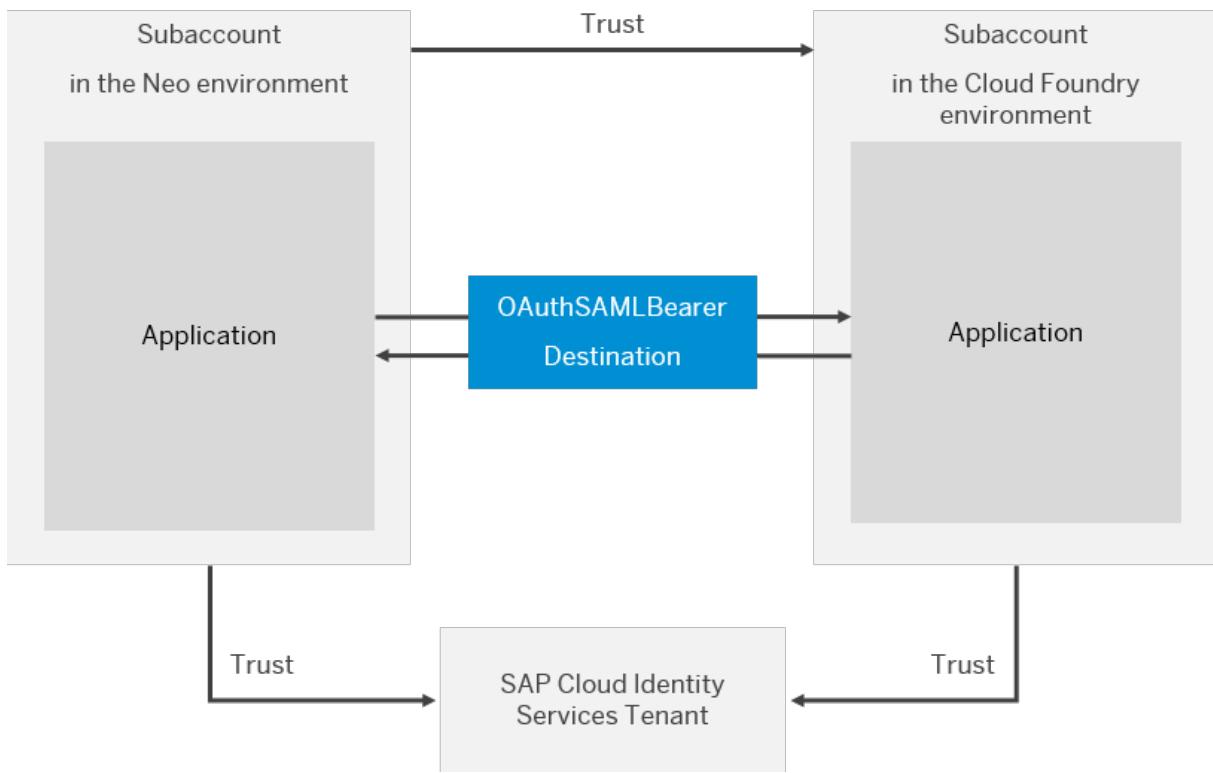
### Related Information

[Propagate User Information Between Applications or Services \[page 508\]](#)

### 7.4.1 Principal Propagation from the Neo to the Cloud Foundry Environment

Enable an application in your subaccount in the Neo environment to access another application in a subaccount in the Cloud Foundry environment without user login (and user interaction) in the second application. For this scenario to work, the two subaccounts need to be in mutual trust, and in trust with the same SAP Cloud Identity Services tenant. The second application will propagate its logged-in user to the first application using an OAuth2SAMLBearer destination.

The graphic below illustrates the overall setup of the scenario.



## Prerequisites

- You have a user account with *Administrator* role in both SAP BTP subaccounts. See [Managing Member Authorizations in the Neo Environment](#).
- You have a custom local service provider configuration (signing keys and certificates, etc.) in your subaccount in the Neo environment. See [Configure the Local Service Provider](#).
- Both accounts have a trust configuration to the same SAP Cloud Identity Services tenant. See:
  - [Identity Authentication Tenant as an Application Identity Provider](#) (for the Neo environment)
  - [Manually Establish Trust and Federation Between SAP Authorization and Trust Management Service and SAP Cloud Identity Services \[page 2224\]](#) (for the Cloud Foundry environment)
- You have developed and deployed both applications, each in the corresponding subaccount.

### ⓘ Note

All configuration steps described in this tutorial are done using the cloud cockpit.

## Requirements to the Application in the Neo Environment

The application is running on a Java Web Tomcat 8 runtime.

In the source code, the application needs to reference the destination that we are about to create as a later step. The sample source code below illustrates a complete servlet working with the destination with name pptest.

```

package com.sap.cloud.samples;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import java.security.KeyStore;
import java.util.List;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSocketFactory;
import javax.net.ssl.TrustManagerFactory;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.sap.core.connectivity.api.authentication.AuthenticationHeader;
import com.sap.core.connectivity.api.authentication.AuthenticationHeaderProvider;
import com.sap.core.connectivity.api.configuration.ConnectivityConfiguration;
import com.sap.core.connectivity.api.configuration.DestinationConfiguration;
@WebServlet("/neotocf")
public class NeoToCF extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private static final Logger LOGGER = LoggerFactory.getLogger(NeoToCF.class);
    private static final String ON_PREMISE_PROXY = "OnPremise";
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.getWriter().println("Served to: " +
        request.getUserPrincipal().getName());
        try {
            // Look up the connectivity configuration API
            Context ctx = new InitialContext();
            ConnectivityConfiguration configuration =
                (ConnectivityConfiguration) ctx
                    .lookup("java:comp/env/connectivityConfiguration");
            // Get destination configuration
            DestinationConfiguration destConfiguration =
                configuration.getConfiguration("pptest");
            if (destConfiguration == null) {
                response.sendError(HttpServletResponse.SC_INTERNAL_SERVER_ERROR,
                    String.format(
                        "Destination %s is not found. Hint: " +
                        "Make sure to have the destination
configured.", "pptest"));
            }
            return;
        }
        AuthenticationHeaderProvider authHeaderProvider =
        (AuthenticationHeaderProvider) ctx
            .lookup("java:comp/env/myAuthHeaderProvider");
        // retrieve the authorization header for OAuth SAML Bearer principal
        propagation
        List<AuthenticationHeader> samlBearerHeader = authHeaderProvider
            .getOAuth2SAMLBearerAssertionHeaders(destConfiguration);
        LOGGER.debug("JWT token from CF XSUAA: " +
        samlBearerHeader.get(1).getValue());
        // get the configured truststore
    }
}

```

```

KeyStore trustStore = destConfiguration.getTrustStore();

// create sslcontext
TrustManagerFactory tmf =
TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());
tmf.init(trustStore);

SSLContext sslcontext = SSLContext.getInstance("TLS");
sslcontext.init(null, tmf.getTrustManagers(), null);
SSLSocketFactory sslSocketFactory = sslcontext.getSocketFactory();

// get the destination URL
String value = destConfiguration.getProperty("URL");
URL url = new URL(value);

// use the sslcontext for url connection
URLConnection urlConnection = url.openConnection();
((HttpsURLConnection) urlConnection).setSSLSocketFactory(sslSocketFactory);

urlConnection.setRequestProperty(samlBearerHeader.get(0).getName(),
samlBearerHeader.get(0).getValue());
urlConnection.setRequestProperty(samlBearerHeader.get(1).getName(),
samlBearerHeader.get(1).getValue());

urlConnection.connect();
response.getWriter().println("Received from CF:");

BufferedReader in = new BufferedReader(new InputStreamReader(
urlConnection.getInputStream()));
String inputLine;
while ((inputLine = in.readLine()) != null)
    response.getWriter().append(inputLine);
in.close();
} catch (Exception e) {
    // Connectivity operation failed
    String errorMessage = e.getMessage();
    LOGGER.error("Connectivity operation failed", e);
    response.sendError(HttpServletRequest.SC_INTERNAL_SERVER_ERROR,
errorMessage);
}
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}
}

```

## Requirements to the Application in the Cloud Foundry Environment

In the Cloud Foundry environment, you need an application following the XSA security model (protected with SAML, UAA service binding using JWT token needed, roles configured in the [xs-security.json](#)).

See:

- [Application Router \[page 409\]](#)
- [Building Roles and Role Collections for Applications \[page 2284\]](#)

### Note

You can use the [XSA Security Sample Application](#) in GitHub (instructions and code) to develop and deploy an application compliant with the above requirements.

## 7.4.1.1 Create a Destination

### Prerequisites

Before you create the required destination, you need to note down a few properties that will be used as values in the destination settings.

1. In the cloud cockpit, navigate to the subaccount in the Cloud Foundry environment.
2. Navigate to the application router.
3. Enter the *Environment Variables* section.
4. Note down somewhere the values of the following properties:
  - clientid
  - clientsecret
  - url

### Context

Connect the first subaccount to the second subaccount by describing the source connection properties in a destination. For more information see [Modeling Destinations](#).

### Procedure

1. Choose the global account and navigate to [Connectivity](#) [Destinations](#).
2. Choose [New Destination](#).
3. In the new destination, provide the following information:

Field	Description
Name	Technical name of the destination. It can be used later on to get an instance of that destination. It must be unique for the global account.
URL	The URL of the protected resource in the Cloud Foundry environment. See <a href="#">Configuring Application URLs</a> .  Example: <code>https://&lt;tenant-specific-route-for-your-business-app&gt;.cfapps.eu10.hana.ondemand.com/</code>

#### ⓘ Note

For the purposes of the example listed in this document, use `ppitest` as value.

Field	Description	
Authentica- tion	OAuth2SAMLBearerAssertion	
Proxy Type	Internet	
Audience	Copy the value of entityID property of the SAML 2.0 metadata representing your subaccount in the Cloud Foundry environment.	
<p><b>→ Tip</b></p> <p>You can open the metadata of the subaccount in the Cloud Foundry environment using the following URL:</p> <pre>https://&lt;your subaccount's subdomain&gt;.authentication.&lt;region host&gt;/saml/metadata</pre> <p>Example:</p> <pre>https://demo.authentication.eu10.hana.ondemand.com/saml/metadata</pre> <p>For the &lt;region host&gt;, see <a href="#">Regions and API Endpoints Available for the Cloud Foundry Environment [page 20]</a>.</p>		
Example of audience/entityID:		
<code>demo.aws-live-eu10</code>		
Client Key	In the cloud cockpit, navigate to the application in the Cloud Foundry environment (▶ <a href="#"><i>&lt;path to your subaccount&gt;</i></a> ▶ <a href="#"><i>Spaces</i></a> ▶ <a href="#"><i>&lt;your space&gt;</i></a> ▶ <a href="#"><i>Applications</i></a> ▶ <a href="#"><i>&lt;your application&gt;</i></a> ▶). Open <a href="#"><i>Environment Variables</i></a> . Copy the value of the <code>clientid</code> property in ▶ <a href="#"><i>VCAP_SERVICES</i></a> ▶ <a href="#"><i>xsuaa</i></a> ▶ <a href="#"><i>credentials</i></a> ▶.	

Field	Description
Token Service URL	<p>Get the <b>token service URL</b> from the SAML 2.0 metadata representing your subaccount in the Cloud Foundry environment. The <b>token service URL</b> is defined in the <code>Location</code> attribute of the element marked as <code>AssertionConsumerService</code>, like this:</p> <pre>&lt;md:AssertionConsumerService Location="<b>&lt;Token Service URL&gt;</b>" Binding="urn:oasis:names:tc:SAML:2.0:bindings:URI" index="1" /&gt;</pre> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <b>→ Tip</b> <p>You can open the metadata of the subaccount in the Cloud Foundry environment using the following URL:</p> <pre>https://&lt;your subaccount's subdomain&gt;.authentication.&lt;region host&gt;/saml/metadata</pre> <p>Example:</p> <pre>https://demo.authentication.eu10.hana.ondemand.com/saml/metadata</pre> <p>For the <code>&lt;region host&gt;</code>, see <a href="#">Regions and API Endpoints Available for the Cloud Foundry Environment [page 20]</a>.</p> </div>
Example of token service URL:	
	<pre>https://demo.authentication.eu10.hana.ondemand.com/oauth/token/alias/demo.aws-live-eu10</pre>
Token Service User	In the cloud cockpit, navigate to the application in the Cloud Foundry environment (▶ <code>&lt;path to your subaccount&gt;</code> ▶ <code>Spaces</code> ▶ <code>&lt;your space&gt;</code> ▶ <code>Applications</code> ▶ <code>&lt;your application&gt;</code> ▶). Open <code>Environment Variables</code> . Copy the value of the <code>clientid</code> property in ▶ <code>VCAP_SERVICES</code> ▶ <code>xsuaa</code> ▶ <code>credentials</code> ▶.
Token Service Password	In the cloud cockpit, navigate to the application in the Cloud Foundry environment (▶ <code>&lt;path to your subaccount&gt;</code> ▶ <code>Spaces</code> ▶ <code>&lt;your space&gt;</code> ▶ <code>Applications</code> ▶ <code>&lt;your application&gt;</code> ▶). Open <code>Environment Variables</code> . Copy the value of the <code>clientsecret</code> property in ▶ <code>VCAP_SERVICES</code> ▶ <code>xsuaa</code> ▶ <code>credentials</code> ▶.
System User	Empty.

- Save the changes.

## 7.4.1.2 Create Trust Between the Subaccounts

### Procedure

- In the cloud cockpit, log on with the Administrator user.
- Save locally the service provider metadata of the subaccount in the Neo environment.

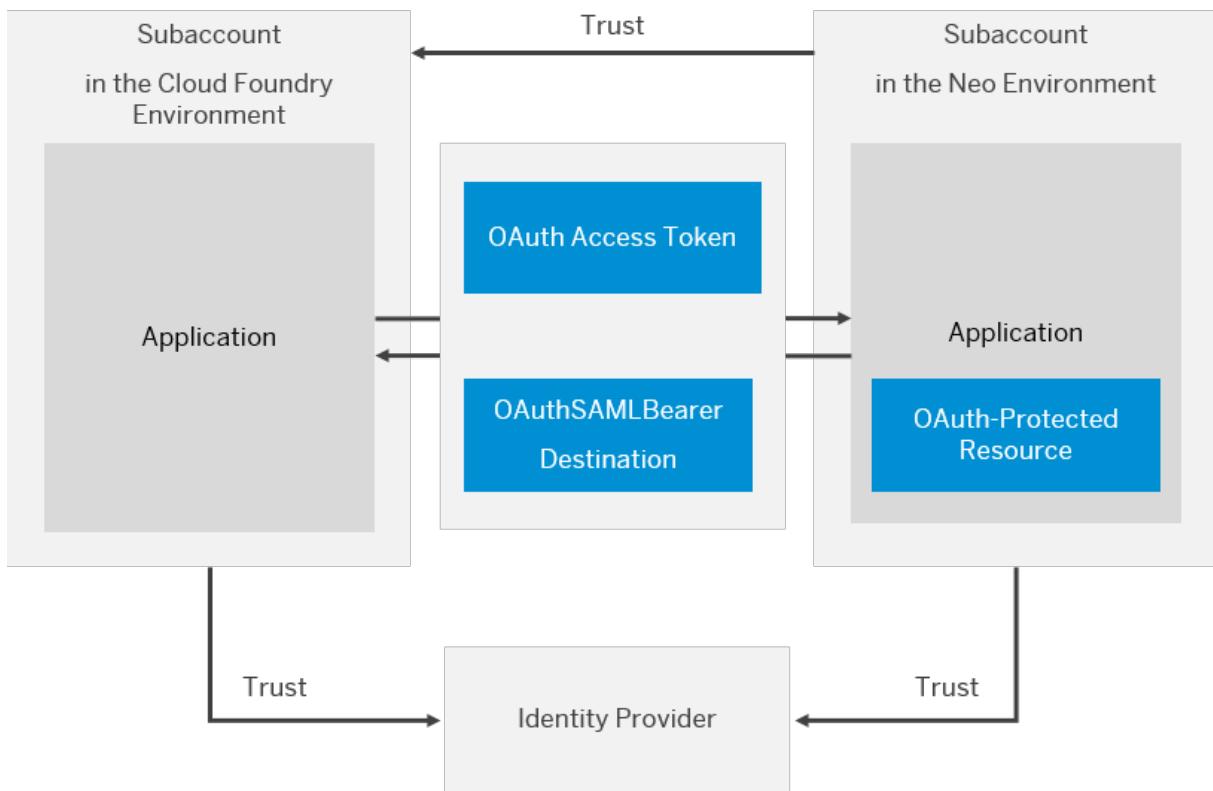
- a. In SAP BTP cockpit, navigate to the subaccount in the Neo environment. See [Navigate in the Cockpit \[page 2142\]](#)
  - b. Navigate to [Security](#) [Trust](#) [Local Service Provider](#)
  - c. Choose [Get Metadata](#) and save the metadata file representing this subaccount.
3. In the subaccount in the Cloud Foundry environment, create trust to the subaccount in the Neo environment.
    - a. Navigate to the subaccount in the Cloud Foundry environment.
    - b. Navigate to [Security](#) [Trust Configuration](#)
    - c. Choose [New Trust Configuration](#).
    - d. In the [Metadata](#) field, choose the [Upload](#) button.
    - e. Upload the service provider metadata file representing the subaccount in the Neo environment.
    - f. Save the new trust configuration.

After this procedure, you can use the security context from the application in the Neo environment to the application in the Cloud Foundry environment. The assigned groups from the Neo environment can be used as role collections in the Cloud Foundry environment.

## 7.4.2 Principal Propagation from the Cloud Foundry to the Neo Environment

Enable an application in your subaccount in the Cloud Foundry environment to access an OAuth-protected application in a subaccount in the Neo environment without user login (and user interaction) in the second application. For this scenario to work, the two subaccounts need to be in mutual trust, and in trust with the same identity provider. The first application will propagate its logged-in user to the second application using an OAuth2SAMLBearer destination.

The graphic below illustrates the overall setup of the scenario.



## Prerequisites

- You have a user account with [Administrator](#) role in both SAP BTP subaccounts. See [Managing Member Authorizations in the Neo Environment](#).
- You have a *custom* local service provider configuration (this means in  $\triangleright$  [cloud cockpit](#)  $\triangleright$  [Security](#)  $\triangleright$  [Trust](#)  $\triangleright$  [Local Service Provider](#)  $\triangleright$  you have chosen  $\triangleright$  [Configuration Type](#)  $\triangleright$  [Custom](#)  $\triangleright$ ) in your subaccount in the Neo environment. See [Configure the Local Service Provider](#).
- Both accounts have a trust configuration to the same identity provider. See:
  - [Configure Trust to the SAML Identity Provider](#) (for the Neo environment)
  - [Establish Trust with Any SAML 2.0 Identity Provider in a Subaccount \[page 2229\]](#) (for the Cloud Foundry environment)
- The application in the Neo environment is protected using OAuth 2.0. See [OAuth 2.0 Service](#).
- The application in the Cloud Foundry environment is bound to an instance of the following services:
  - [Destination Service](#). See [Create and Bind a Destination Service Instance](#).
  - [xsuaa](#)
- You have deployed both applications, each in the corresponding subaccount.

### ① Note

All configuration steps described in this tutorial are done using the cloud cockpit.

## Contents

- [Create Trust Between the Subaccounts \[page 3104\]](#)
- [Create an OAuth Client \[page 3105\]](#)
- [Create a Destination \[page 3106\]](#)

### 7.4.2.1 Create Trust Between the Subaccounts

Exchange keys and certificates between the subaccounts, and configure trust between them. This will enable the subaccounts to communicate using HTTP destinations.

#### Procedure

1. In the cloud cockpit, log on with the Administrator user.
2. Save locally the identifying X509 certificate of the subaccount in the Cloud Foundry environment.
  - a. In SAP BTP cockpit, navigate to the subaccount in the Cloud Foundry environment.
  - b. Navigate to [Connectivity](#) [Destinations](#).
  - c. Choose [Download Trust](#) and save the X509 certificate identifying this subaccount.
3. In the subaccount in the Neo environment, create trust to the subaccount in the Cloud Foundry environment.
  - a. Navigate to the subaccount in the Neo environment.
  - b. Navigate to [Security](#) [Trust](#) [Application Identity Provider](#).
  - c. Choose [Add Trusted Identity Provider](#) and configure the new trust configuration settings:

- In the [Name](#) field, enter the following:

<your Cloud Foundry domain host>/<Cloud Foundry subaccount ID>

→ Tip

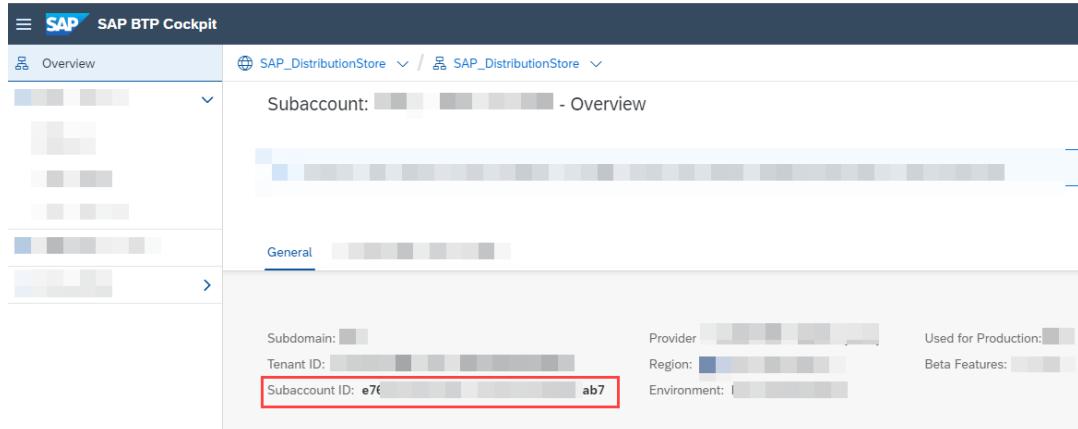
You can view the **Cloud Foundry domain host** in [cockpit](#) [<your global account>](#) [<your subaccount>](#) [<your space>](#) [Routes](#).

The screenshot shows the SAP BTP Cockpit interface with the title bar "SAP BTP Cockpit". The left sidebar has a "Routes" item selected. The main area displays a table titled "Space: DistributionStore - Routes" with 11 entries. The columns are: Host Name, Path, Domain, Mapped Applications, Route Service Instance, and Route Service. One row in the table is highlighted with a red border, showing the domain "cfapps.eu10.hana.ondemand.com".

Host Name	Path	Domain	Mapped Applications	Route Service Instance	Route Service
[redacted]	-	cfapps.eu10.hana.ondemand.com			
[redacted]	-	cfapps.eu10.hana.ondemand.com			
[redacted]	-	cfapps.eu10.hana.ondemand.com			
[redacted]	-	cfapps.eu10.hana.ondemand.com			

### → Tip

You can view the **subaccount ID** in ► [cockpit](#) ► <your global account> ► <your subaccount> ► [Overview](#).



The screenshot shows the SAP BTP Cockpit interface. In the top navigation bar, it says "SAP BTP Cockpit". Below that, there's a breadcrumb navigation: "Overview" > "SAP\_DistributionStore" > "SAP\_DistributionStore". The main content area is titled "Subaccount: [REDACTED] - Overview". Under the "General" tab, there are several fields: "Subdomain: [REDACTED]", "Provider: [REDACTED]", "Used for Production: [REDACTED]", "Tenant ID: [REDACTED]", "Region: [REDACTED]", "Beta Features: [REDACTED]", and "Subaccount ID: e7c [REDACTED] ab7". The "Subaccount ID" field is highlighted with a red border.

- In the *Signing Certificate* field, enter the X509 certificate of the Cloud Foundry account.

### ⓘ Note

Make sure you remove the BEGIN CERTIFICATE and END CERTIFICATE parts.

- Mark the *Only for IDP-initiated SSO* option.
  - d. Save the new trust configuration.

## 7.4.2.2 Create an OAuth Client

You need an OAuth client to get an access token for the OAuth-protected resources in the application in the Neo environment.

### Context

For more information about working with OAuth clients, see [Create an OAuth Client \[page 3105\]](#).

### Procedure

1. In the cockpit, navigate to the subaccount in the Neo environment.
2. Navigate to ► [Security](#) ► [OAuth](#) ► [Clients](#), and choose [Register New Client](#).
3. Create an OAuth client with the following configuration settings:

- *Name* - the OAuth client name. You will need to provide this name as value of the *Token Service User* property of the destination below.
  - *Authorization Grant* - choose the *Authorization Code* option.
  - Mark the *Confidential* option, and provide a secret (password).
4. Save the client.

#### ⓘ Note

When creating the required **OAuthSAMLBearer destination** later, you will need the following information from the OAuth client:

- *ID*
- *Secret*

### 7.4.2.3 Create a Destination

#### Context

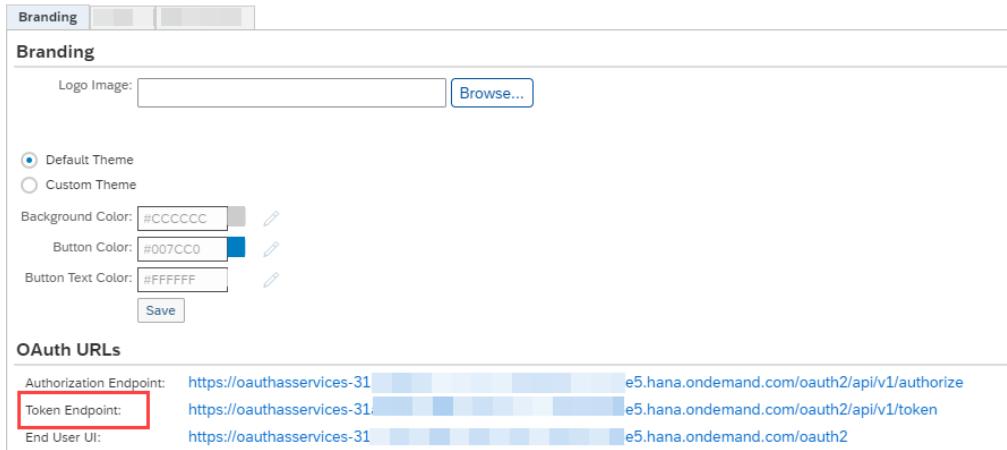
Connect the two subaccounts by describing the connection properties in a destination. For more information, see [Modeling Destinations](#).

#### Procedure

1. Choose the subaccount in the Cloud Foundry environment, and navigate to [Connectivity](#) [Destinations](#).
2. Choose [New Destination](#).
3. In the new destination, provide the following information:

Field	Description
Name	Technical name of the destination. It can be used later on to get an instance of that destination. It must be unique for the global account.
Description	Free-text description.
Type	HTTP
URL	The URL of the protected resource in the Neo environment. Example: <code>https://myneoapp.hana.ondemand.com/myprotectedresource/</code>

Field	Description
Authentica- tion	OAuth2SAMLBearerAssertion
Proxy Type	Internet
Audience	The value of the local service provider name in the subaccount in the Neo environment.  Copy the value from <a href="#">cockpit</a> <a href="#">&lt;your Neo subaccount&gt;</a> <a href="#">Security</a> <a href="#">Trust</a> <a href="#">Local Service Provider</a> <a href="#">Local Service Provider Name</a> .
Trust Management	<p> <a href="#">Manage Local Provider Settings</a> for </p> <p>Configuration Type: Custom</p> <p>Local Provider Name: <input type="text" value="https://eu1.hana.ondemand.com/neotest"/> </p> <p>Signing Key: <input type="text" value="MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKwggSlAgEAAoIBAQDMskSvgfAsOfzxC+7ld3WRwPdnGBUx+yy0bfNN9ZzwcwJy2wzC2Kd8FlcHFLNAb1BxxK3ru0XILnpW7BAgTRyrSP0e6UCHsFJ26KN49vvixFr1CeJ9USy20K7rw+uH66f69pL+QK2HVV6gJZPj3paO60cvz19/HURbs+5daeTRqGMhzEzg+6X9ETQCesQUFE9JH/4L3wnsF6gvzRYFO79vsPbzr93vPqCAT4RMTi2NlwJVwM0hXF9SS6rl0DXZExOwmo7NccFxlvsx0XlcfIF9iunnvn9cReJscn708vXnYTNI+XOnGXAx09CWC/061P0In6VnXeGy-TbA"/></p> <p>Signing Certificate: <input type="text" value="MIIDdTCCAI2gAwIBAgIEWhnh8DANBgkqhkiG9w0BAQUFADBrMQswCQYDVQQGEwJERTEPMA0GA1UEChMGU0fQIEFHMRswGQYDVQQLExJTQVAgQ2xvdWogUGxhdGZvcm0xLJA5BgNVBAMTJWWh0dBzol8vZXUxLmhnbmEub25kZW1hbmQuY29tL25lb3Rlc30wHhcNMNTgwMjzMDg0NDAwWjBrMQswCQYDVQQGEwJERTEPMA0GA1UEChMGU0fQIEFHMRswGQYDVQQLExJTQVAgQ2xvdWQoLjGxhdGZvcm0xIiAsRaNVRAMTJWWh0dhHz0iRvZXUxLmhnbmEub25kZW1hbmQ"/></p> <p>Signing Certificate Validity: from Fri Feb 23 08:44:00 UTC 2018 to Wed Feb 23 08:44:00 UTC 2028</p> <p>Principal Propagation: <input type="text" value="Enabled"/></p> <p>Force Authentication: <input type="text" value="Disabled"/></p> <p><a href="#">Edit</a> <a href="#">Get Metadata</a></p>
Client Key	The ID of the OAuth client for the application in the Neo environment.

Field	Description
Token Service URL	Copy the value of <i>Token Endpoint</i> from the following place:  cockpit > <your Neo subaccount> > Security > OAuth > Branding.
OAuth Settings	 <p>The screenshot shows the 'Branding' configuration page in the SAP cockpit. It includes sections for 'Logo Image' (with a 'Browse...' button), theme selection ('Default Theme' or 'Custom Theme'), and color customization for 'Background Color' (#CCCCCC), 'Button Color' (#007CC0), and 'Button Text Color' (#FFFFFF). A 'Save' button is at the bottom. Below this is the 'OAuth URLs' section, which lists the 'Authorization Endpoint' (https://oauthasservices-31.e5.hana.ondemand.com/oauth2/api/v1/authorize), 'Token Endpoint' (https://oauthasservices-31.e5.hana.ondemand.com/oauth2/api/v1/token), and 'End User UI' (https://oauthasservices-31.e5.hana.ondemand.com/oauth2).</p>
Token Service User	The ID of the OAuth client for the application in the Neo environment.
Token Service Password	The secret from the OAuth client.
System User	Empty.
authnContextClassRef	<i>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</i>
nameIdFormat	<i>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</i> if the user ID will be propagated to the Neo application or <i>nameIdFormat = urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</i> if the user email will be propagated to the Neo application.

Destination Configuration

<b>*Name:</b>	CFtoNeo	<b>Additional Properties</b>
<b>Type:</b>	HTTP	nameIdFormat <input type="button" value="..."/> urn:oasis:names:tc:SAML:2.0:nameid-format:unspecified <input type="button" value="Delete"/>
<b>Description:</b>	Communication between the two environments (CF -> neo)	
<b>*URL:</b>	<a href="https://myneoadapp.hana.ondemand.com/myprotectedresource/">https://myneoadapp.hana.ondemand.com/myprotectedresource/</a>	
<b>Proxy Type:</b>	Internet	authnContextClass <input type="button" value="..."/> jknhjlnh <input type="button" value="Delete"/>
<b>Authentication:</b>	OAuth2SAMLBearerAssertion	
<b>*Audience:</b>	<a href="https://hana.ondemand.com/myneosubaccount">https://hana.ondemand.com/myneosubaccount</a>	
<b>*Client Key:</b>	*****	
<b>*Token Service URL:</b>	<a href="https://oauth2services-myneosubaccount.hana.ondemand.co">https://oauth2services-myneosubaccount.hana.ondemand.co</a>	
<b>Token Service User:</b>	937f7df2-99cc-3983-a191-b16e88203c0b	
<b>Token Service Password:</b>	*****	
<b>System User:</b>		

Use default JDK truststore

- Save the changes.

## 7.5 Data Protection and Privacy

Data protection is associated with numerous legal requirements and privacy concerns. In addition to compliance with general data protection and privacy acts, it's necessary to consider compliance with industry-specific legislation in different countries/regions.

SAP provides specific features and functions to support compliance with regard to relevant legal requirements, including data protection. SAP does not give any advice on whether these features and functions are the best method to support company, industry, regional, or country/region-specific requirements. Furthermore, this information shouldn't be taken as advice or a recommendation regarding additional features that would be required in specific IT environments. Decisions related to data protection must be made on a case-by-case basis, taking into consideration the given system landscape and the applicable legal requirements.

### ⓘ Note

SAP does not provide legal advice in any form. SAP software supports data protection compliance by providing security features and specific data protection-relevant functions. In many cases, compliance with applicable data protection and privacy laws will not be covered by a product feature. Definitions and other terms used in this document are not taken from a particular legal source.

### ⚠ Caution

The extent to which data protection is supported by technical means depends on secure system operation. Network security, security note implementation, adequate logging of system changes, and appropriate usage of the system are the basic technical requirements for compliance with data privacy legislation and other legislation.

## Generic Fields

You also need to make sure that no personal data enters the system in an uncontrolled or non-purpose related way, for example, in free-text fields, or customer extensions.

### Caution

Don't enter personal data in fields, such as global and subaccount names, Cloud Foundry orgs and spaces names, database names, and tenant names. Personal data in such fields can't be detected and removed during data deletion requests. Even if such cases could be identified, removing such data may lead to disruptive and unexpected behavior in our services. The previous list of example fields isn't complete.

## SAP BTP

This documentation covers personal data relating to SAP BTP accounts and data stored in databases by SAP BTP. SAP BTP offers a number of capabilities, that is, services, buildpacks, application, and so on. Here we cover the core platform. For more information about data protection and privacy for capabilities you've purchased, see the data protection and privacy documentation for those capabilities.

To view the services consumed by a global account:

1. Navigate to the global account to which you'd like to view members.  
For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).
2. In the navigation area, choose *Entitlements*.

This documentation is written with the data protection officer of a company in mind. The processes described here may be required for a data protection officer or an administrator of the user accounts for your tenants or even business users of the tenants. In particular the processes for business users are described here so that you in your role of data protection officer or account administrator can communicate them to your business users if required.

Users are stored in the platform identity provider.

- Global account users are stored in platform identity provider or a tenant of SAP Cloud Identity Services.
- Platform users are stored in platform identity provider, a tenant of SAP Cloud Identity Services, or your own identity provider.
- Business users are stored in a tenant of SAP Cloud Identity Services or your own identity provider.

## Related Information

[Glossary for Data Protection and Privacy \[page 3111\]](#)

[Change Logging and Read-Access Logging \[page 3113\]](#)

[Personal Data Record \[page 3113\]](#)

[Deletion \[page 3114\]](#)

[Consent \[page 3120\]](#)

## 7.5.1 Glossary for Data Protection and Privacy

The following terms are general to SAP products. Not all terms may be relevant for SAP BTP.

Term	Definition
<b>Blocking</b>	A method of restricting access to data for which the primary business purpose has ended.
<b>Business purpose</b>	The legal, contractual, or in other form justified reason for the processing of personal data to complete an end-to-end business process. The personal data used to complete the process is predefined in a purpose, which is defined by the data controller. The process must be defined before the personal data required to fulfill the purpose can be determined.
<b>Consent</b>	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.
<b>Data subject</b>	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.
<b>Deletion</b>	Deletion of <b>personal data</b> so that the data is no longer available.
<b>End of business</b>	Defines the end of active business and the start of residence time and retention period.
<b>End of purpose (EoP)</b>	The point in time when the processing of a set of personal data is no longer required for the primary business purpose, for example, when a contract is fulfilled. After the EoP has been reached, the data is blocked and can only be accessed by users with special authorizations (for example, tax auditors).
<b>End of purpose (EoP) check</b>	A method of identifying the point in time for a data set when the processing of <b>personal data</b> is no longer required for the primary <b>business purpose</b> . After the <b>EoP</b> has been reached, the data is <b>blocked</b> and can only be accessed by users with special authorization, for example, tax auditors.

Term	Definition
<b>Personal data</b>	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.
<b>Purpose</b>	The information that specifies the reason and the goal for the processing of a specific set of personal data. As a rule, the purpose references the relevant legal basis for the processing of personal data.
<b>Residence period</b>	The period of time between the end of business and the end of purpose (EoP) for a data set during which the data remains in the database and can be used in case of subsequent processes related to the original purpose. At the end of the longest configured residence period, the data is blocked or deleted. The residence period is part of the overall retention period.
<b>Retention period</b>	The period of time between the end of the last business activity involving a specific object (for example, a business partner) and the deletion of the corresponding data, subject to applicable laws. The retention period is a combination of the residence period and the blocking period.
<b>Sensitive personal data</b>	<p>A category of personal data that usually includes the following type of information:</p> <ul style="list-style-type: none"> <li>• Special categories of personal data, such as data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetic data, biometric data, data concerning health or sex life or sexual orientation.</li> <li>• Personal data subject to professional secrecy</li> <li>• Personal data relating to criminal or administrative offenses</li> <li>• Personal data concerning insurances and bank or credit card accounts</li> </ul>

Term	Definition
<b>Technical and organizational measures (TOM)</b>	<p>Some basic requirements that support data protection and privacy are often referred to as technical and organizational measures (TOM). The following topics are related to data protection and privacy and require appropriate TOMs, for example:</p> <ul style="list-style-type: none"><li>• <b>Access control:</b> Authentication features</li><li>• <b>Authorizations:</b> Authorization concept</li><li>• <b>Read access logging</b></li><li>• <b>Transmission control / Communication security</b></li><li>• <b>Input control / Change logging</b></li><li>• <b>Availability control</b></li><li>• <b>Separation by purpose:</b> Is subject to the organizational model implemented and must be applied as part of the authorization concept.</li></ul>

## 7.5.2 Change Logging and Read-Access Logging

Change logging records changes to personal data, while read-access logging records access to sensitive personal data. You may be required to gather this information for auditing purposes or legal requirements.

Use the appropriate API to retrieve logs.

- For the Cloud Foundry, see [Audit Log Retrieval API Usage for Subaccounts in the Cloud Foundry Environment \[page 2503\]](#).

### ⓘ Note

For any applications you develop, you must ensure they include logging functions. SAP BTP does not provide audit logging functions for custom developments.

Logs are deleted regularly.

## 7.5.3 Personal Data Record

A personal data record is a collection of data relating to a data subject. A data privacy specialist may be required to provide such a record or an application may offer a self-service.

To see the personal data that is used for membership management within SAP BTP, access the cloud cockpit.

1. Navigate to the global account to which you'd like to view members.  
For more information, see [Navigate to Orgs and Spaces \[page 2433\]](#).
2. In the navigation area, choose *Members*.

To see the personal data that is used for application logging within SAP BTP, access the cloud cockpit.

For more information, see [Using Logs in the Cockpit](#) or [Analyze Logs from the Cockpit](#) in the Application Logging service documentation.

If you do not use your own identity provider for identity federation, you can view the profiles available in SAP Cloud Identity Services - Identity Authentication.

For more information, see [Information Report](#) in the SAP Cloud Identity Services - Identity Authentication documentation.

For SAP BTP Cloud Foundry environment, the User Account and Authentication service creates shadow users to issue tokens for their corresponding users.

For more information about viewing shadow users and their attributes, see the [User Management \(SCIM\) API](#) on [SAP Business Accelerator Hub](#).

For all other services, which persist data, such as databases or document services, retrieve the data you stored with the same APIs, protocols, or languages you used to store the data.

To view the services used in a global account, choose [Entitlements](#) in the navigation area.

## 7.5.4 Deletion

The processing of personal data is subject to applicable laws related to the deletion of this data when the specified, explicit, and legitimate purpose for processing this personal data has expired. If there is no longer a legitimate purpose that requires the retention and use of personal data, it must be deleted.

When deleting data in a data set, all referenced objects related to that data set must be deleted as well. Industry-specific legislation in different countries also needs to be taken into consideration in addition to general data protection laws. After the expiration of the longest retention period, the data must be deleted.

When accounts expire, we delete your data barring legal requirements that SAP retains your data. If your organization has separate retention requirements, you are responsible for saving this data before we terminate your account.

- For trial accounts in the Cloud Foundry environment, your account expires after 365 days.
- Productive accounts expire based on the terms of your contract.

To deactivate or delete users, see [Erasure](#) in the SAP Cloud Identity Services documentation.

For SAP BTP Cloud Foundry environment, the User Account and Authentication service creates shadow users to issue tokens for their corresponding users. Data privacy regulations or policies may require you to delete this data, for example, when the user has left your organization.

For more information about deleting shadow users, see:

- [Delete Shadow Users for Data Protection and Privacy Using the Cockpit \[page 3115\]](#)
- [Delete Shadow Users for Data Protection and Privacy Using APIs \[page 3118\]](#)

### → Remember

When you delete a user at the Cloud Foundry level, SAP BTP doesn't delete the corresponding shadow user at the subaccount level.

When you delete a user at the subaccount level, ensure that you delete the user at the Cloud Foundry level too. Otherwise, if the Cloud Foundry user logs on again, the system automatically creates the corresponding user at the subaccount level.

So ensure that you delete the shadow users on all levels.

For all other services which persist data, you can retrieve the data you stored with the same APIs, protocols, or languages which you used to store the data.

To view the services used in a global account, choose *Entitlements* in the navigation area.

SAP BTP Data Retention Manager is a service available for the Cloud Foundry environment that helps you to identify data subjects for deletion as well as maintain rules for residence and retention.

For more information, see [SAP BTP Data Retention Manager](#).

We maintain backups of the data for disaster recovery. When your account is deleted, we may have this data in our backup system for the length of our backup cycle.

#### ⓘ Note

If your data is stored outside SAP BTP, we cannot guarantee that your data does not get reintegrated if you are pushing such data to our systems. You are responsible for terminating such integrations.

We cannot restore data you have in your local system.

### 7.5.4.1 Delete Shadow Users for Data Protection and Privacy Using the Cockpit

Data privacy regulations or policies may require you to delete this data, for example, when the user has left your organization. SAP BTP cockpit offers an application to select and delete shadow users.

#### Prerequisites

You have the required authorizations.

For more information, see [Security Administration: Managing Authentication and Authorization \[page 2202\]](#) or [Role Collections and Roles in Global Accounts, Directories, and Subaccounts \[page 107\]](#).

#### Context

#### ⓘ Note

When handling personal data, consider the legislation in the various countries and regions where your organization operates. After the data has passed the end of purpose, regulations might require you to delete the data. For more information on data protection and privacy, see the related link.

The User Account and Authentication service stores user-related data records in the form of shadow users. The UAA uses the information of the shadow users to issue tokens that refer to the specific user. If automatic shadow user creation is enabled, the UAA creates the shadow users when the user authenticates. Otherwise, the UAA creates the shadow user as soon as you assign the user a role collection. These conditions apply to platform users and business users. For more information about shadow users, see the Cloud Foundry documentation.

### ⓘ Note

You can also delete users using the User Management (SCIM) API.

For more information, see [Delete Shadow Users for Data Protection and Privacy Using APIs \[page 3118\]](#).

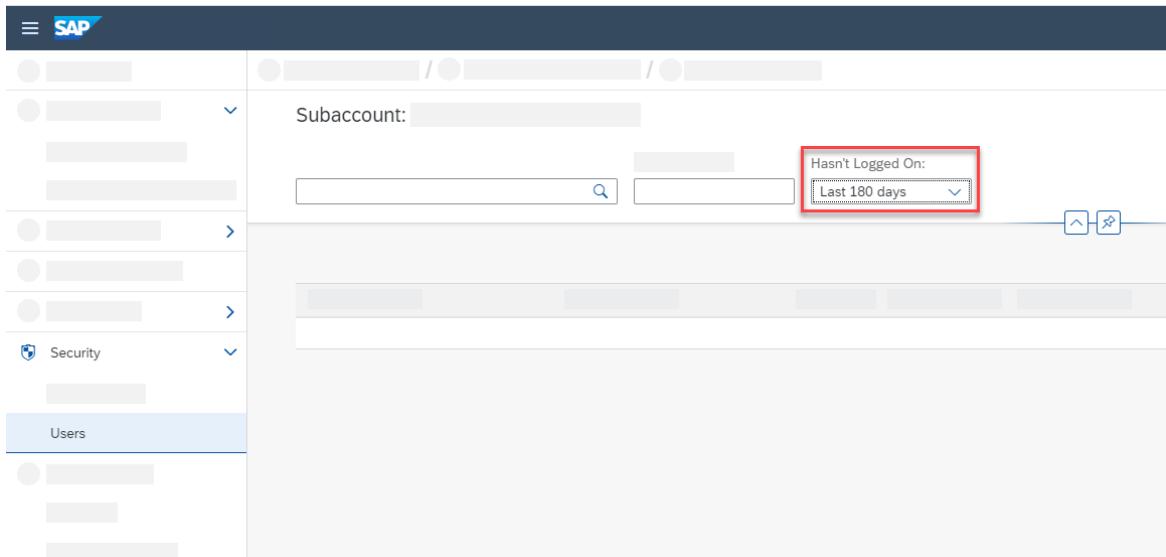
## Procedure

1. Open the SAP BTP cockpit.
2. Go to your global account, directory, or subaccount (see [Navigate in the Cockpit \[page 2142\]](#)).
3. Choose **Security > Users**.

### ⓘ Note

For directories, choose **Users**. There is no **Security** menu item.

4. From the list of users, select the users to delete.



The screenshot shows the SAP BTP cockpit interface. On the left, there's a sidebar with 'Subaccount' dropdowns and a 'Security' section. The main area is titled 'Users' and lists several user entries. At the top right of the main area, there are two input fields: one for 'Hasn't Logged On' with a dropdown menu showing 'Last 180 days' selected, and another for 'Last 30 days'. Below these filters, there are search and refresh buttons. The entire interface has a light gray background with blue and white UI elements.

Filter for Users Who Haven't Logged On

### → Recommendation

Users, who have left your organization, haven't logged on recently. Use the **Hasn't Logged On** filter option to find users who haven't logged on in the last 180 days.

The last logon date you filter for varies from situation to situation. The filter value must meet the following requirements:

- Be long enough that the filter doesn't catch users who are simply on vacation or don't work regularly in the system.
- Not be so long that the filter no longer meets your data protection and privacy requirements.

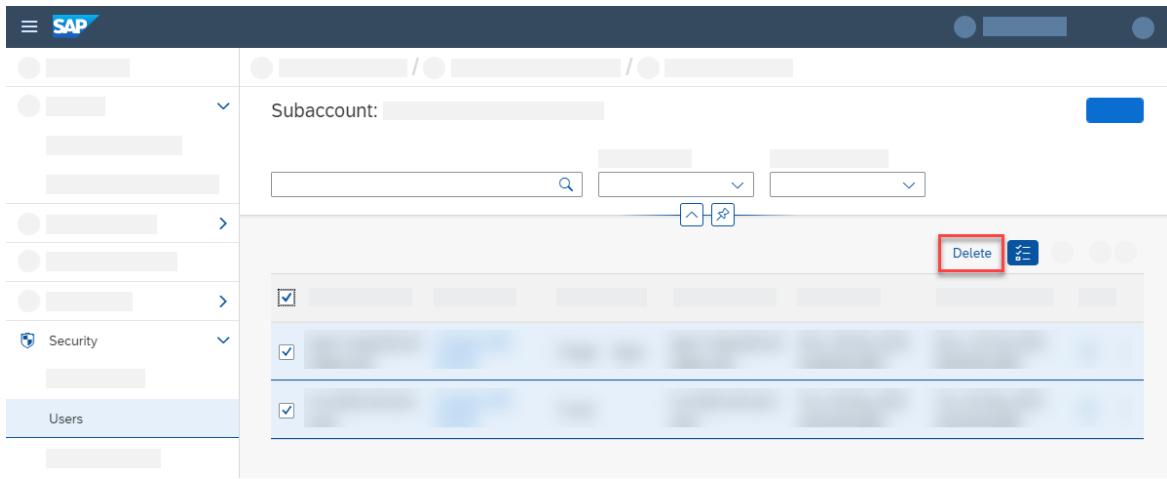
Think about users on parental leave or on a longer leave of absence, too.

### ⚠ Caution

You can't undo the deletion of a user. When you delete a user, you delete any direct assignments of role collections to that user. You also potentially invalidate any application referencing that user.

For example, an application saves some data relating to a user. When you delete the user, that data points to an ID that no longer exists. You can recreate the user, but the new user has a different ID, even though other attributes, such as e-mail, first name, and last name, are identical.

5. Delete the selected users.



## Related Information

[Delete Users \[page 2272\]](#)

[Data Protection and Privacy \[page 3109\]](#)

[Switch Off Automatic Creation of Shadow Users \[page 2264\]](#)

[Shadow Users in the Cloud Foundry Documentation](#) ↗

## 7.5.4.2 Delete Shadow Users for Data Protection and Privacy Using APIs

Data privacy regulations or policies may require you to delete this data, for example, when the user has left your organization. To delete shadow users using APIs, set up access to the API and then use the SCIM REST APIs to retrieve and delete the users.

### Prerequisites

The security administrator must have the following scopes:

- `xs_user.read`
- `xs_user.write`

### Context

#### ⓘ Note

When handling personal data, consider the legislation in the various countries and regions where your organization operates. After the data has passed the end of purpose, regulations might require you to delete the data. For more information on data protection and privacy, see the related link.

The User Account and Authentication service stores user-related data records in the form of shadow users. The UAA uses the information of the shadow users to issue tokens that refer to the specific user. If automatic shadow user creation is enabled, the UAA creates the shadow users when the user authenticates. Otherwise, the UAA creates the shadow user as soon as you assign the user a role collection. These conditions apply to platform users and business users. For more information about shadow users, see the Cloud Foundry documentation.

#### ⓘ Note

Administrators can also delete users using the SAP BTP cockpit. For more information, see [Delete Users \[page 2272\]](#).

### Procedure

1. Enable API access to your subaccount.

For more information, see [Get Access to the APIs \[page 2417\]](#).

2. Use the `GET` method for the User Management (SCIM) API of the SAP Authorization and Trust Management service to get a list of users.

For more information about the User Management (SCIM) API, see <https://api.sap.com/package/authtrustmgmnt> on SAP Business Accelerator Hub.

The API returns the list of users in JSON format.

3. Use your own tools to sort and identify the users in the JSON response to delete.

#### → Recommendation

Users, who have left your organization, haven't logged on recently. Use the `lastLogonTime` parameter to find users who haven't logged on in the last 180 days. The `lastLogonTime` parameter is in UNIX time.

The last logon time you filter for varies from situation to situation. The filter value must meet the following requirements:

- Be long enough that the filter doesn't catch users who are simply on vacation or don't work regularly in the system.
- Not be so long that the filter no longer meets your data protection and privacy requirements.

Account for users on parental leave or on a longer leave of absence, too.

#### ⚠ Caution

You can't undo the deletion of a user. When you delete a user, you delete any direct assignments of role collections to that user. You also potentially invalidate any application referencing that user.

For example, an application saves some data relating to a user. When you delete the user, that data points to an ID that no longer exists. You can recreate the user, but the new user has a different ID, even though other attributes, such as e-mail, first name, and last name, are identical.

4. Use the `DELETE` method and the list of user IDs to delete the shadow users one at a time.

For each user, call the `Users` endpoint with the delete method and include the user ID in the path. For example:

```
curl --location --request  
DELETE 'https://api.authentication.eu20.hana.ondemand.com/Users/a0a67e6f-  
c4b5-41e5-b871-6fa181d46599' \  
--header 'Content-Type: application/x-www-form-urlencoded' \  
--header 'Accept: application/json' \  
--header 'Authorization: bearer eyJqdGkiOiJmZmZl...'
```

For more information about the User Management (SCIM) API, see <https://api.sap.com/package/authtrustmgmnt> on SAP Business Accelerator Hub.

## Related Information

[Data Protection and Privacy \[page 3109\]](#)

<https://docs.cloudfoundry.org/uaa/uaa-concepts.html#%23shadow>

[Switch Off Automatic Creation of Shadow Users \[page 2264\]](#)

[Delete Shadow Users for Data Protection and Privacy Using the Cockpit \[page 3115\]](#)

## 7.5.5 Consent

SAP BTP supports you in collecting and managing the consent of data subjects in the following ways:

SAP Cloud Identity Services provides tools to manage privacy policies and terms of use agreements.

For more information, see [Configuring Privacy Policies](#) and [Configuring Terms of Use](#) in the SAP Cloud Identity Services documentation.

See also [Consent](#) in the SAP Cloud Identity Services documentation.

## 7.6 Security in the Kyma Environment

Security in SAP BTP, Kyma runtime comprises the following pillars:

- [Kyma Security Concepts \[page 3120\]](#)
- [Secure Administration and Operations in the Kyma Environment \[page 3132\]](#)
- [Secure Development in the Kyma Environment \[page 3126\]](#)

Go to the respective topics to learn more about the security features and configurations they propose.

### 7.6.1 Kyma Security Concepts

SAP BTP, Kyma runtime takes various security measures regarding your cluster and its underlying infrastructure. Benefit from the security setup that SAP BTP, Kyma runtime provides.

#### Cloud Infrastructure

All components required to run Kyma, namely Gardener, Kyma Control Plane, and a Kyma cluster, are deployed on SAP-managed IaaS provider accounts. SAP has strict guidelines to secure IaaS provider accounts and the resources deployed into them. The secure configuration is monitored regularly, and the following preventive controls are in place to revert critical configurations that weaken the security of the IaaS provider account:

#### Authentication

- A strict password policy is in place.
- Multi-factor authentication is enabled.

#### Authorization

- Permissions are assigned based on role-based access control (RBAC).
- The principle of least privilege is followed.

## **Storage**

- Storage services are not publicly accessible.
- Encryption using Advanced Encryption Standard (AES) with a key length of at least 256 is used.

## **Kubernetes Control Plane**

Kyma uses Gardener as its managed Kubernetes offering. With Gardener, the control plane of a Kubernetes cluster is hosted as a set of Pods in the Gardener environment. See the overview of the [Gardener architecture](#). In order to harden the Kubernetes control plane as well as the Kubernetes components on the worker nodes, Gardener sets up the cluster in accordance with the Defense Information Systems Agency (DISA) Security Technical Implementation Guides (STIGs) requirements ([V1R6](#)).

The Kubernetes control plane fulfills the following DISA STIGs requirements:

### **General**

- A kubectl version higher than 1.12.9 is used on the control plane (V-242396).
- The Kubernetes DynamicAuditing (V-242398) and DynamicKubeletConfig (V-242399) are disabled.
- Kubernetes manifests (V-242405), component manifests (V-242444), configuration files (V-242446), component PKI (V-242451), and `kubeadm.conf` (V-242454) are owned by root.
- The Kubernetes manifest files have restrictive file permissions following the least privileges principle (V-242408).
- Kubernetes doesn't store Secrets as environment variables (V-242415).
- The role-based access control (RBAC) is activated to control access (V-242435).
- Kubernetes is kept up to date with the latest patches, service packs, and hot fixes. A patch management process is in place (V-242443).
- The Kubernetes `kubeadm.conf` (V-242455), `admin.conf` (V-242460), and PKI CRT (V-242466) have restrictive file permissions.
- Access to the directory containing the Kubernetes PKI files is restricted (V-242467).
- Kubernetes endpoints use the approved organizational certificate and key pair to protect information in transit (V-245544).

### **API Server**

The Kubernetes kube-apiserver fulfills the following DISA STIGs requirements:

- TLS 1.2, at a minimum, is used to protect the confidentiality of sensitive data during electronic dissemination (V-242378).
- The Alpha APIs are disabled (V-242400).
- The insecure bind address (V-242388) and insecure port (V-242386) are not set, and a secure API server port is configured (V-242389).
- The approved SSL Certificate Authority is set (V-242419).
- Only ports, protocols, and services (PPS) that adhere to the Ports, Protocols, and Services Management Category Assurance List (PPSM CAL) are configured (V-242410).
- Approved cipher suites are used (V-242418).
- A certificate for communication is used (V-242422).

- The `ValidatingAdmissionWebhook` is enabled (V-242436).
- Appropriate timeouts are set to limit attack surface (V-242438).
- Restrictive file permissions for the configuration files of the Control Plane services are used (V-242458).
- Only secure encryption protocols are used. SSL, TLS 1.0, and 1.1 are prohibited. (V-242468).

## Kubernetes Scheduler

The Kubernetes scheduler, namely, `kube-scheduler` fulfills the following DISA STIGs requirements:

- TLS 1.2, at a minimum, is used to protect the confidentiality of sensitive data during electronic dissemination (V-242377).
- A secure binding to localhost (127.0.0.1) is used (V-242384).
- Only ports, protocols, and services (PPS) that adhere to the Ports, Protocols, and Services Management Category Assurance List (PPSM CAL) are configured (V-242411).

## Controller Manager

The Kubernetes Controller Manager fulfills the following DISA STIGs requirements:

- TLS 1.2, at a minimum, is used to protect the confidentiality of sensitive data during electronic dissemination (V-242376).
- Unique service accounts for each work payload are used (V-242381).
- A secure binding to localhost (127.0.0.1) is used (V-242385).
- Profiling is disabled (V-242409).
- The SSL Certificate Authority is set (V-242421).
- Only ports, protocols, and services (PPS) that adhere to the Ports, Protocols, and Services Management Category Assurance List (PPSM CAL) are configured (V-242412).

## etcd

The Kubernetes etcd fulfills the following DISA STIGs requirements:

- The usage of self-signed certificates for client-to-server and server-to-server communication is prohibited (V-242379, V-242380).
- Only ports, protocols, and services (PPS) that adhere to the Ports, Protocols, and Services Management Category Assurance List (PPSM CAL) are configured (V-242413).
- Certificate-base client authentication is enabled (V-242423), (V-242426).
- TLS encrypted client-to-server and server-to-server communication is configured (V-242427, V-242432, V-242433). The communication between the API server and etcd is encrypted with TLS (V-242431).
- Approved and valid certificates for the TLS encrypted connection between the API server and etcd are used (V-242428, V-242430). An SSL certificate authority is configured (V-242429).
- Files are owned by `etcd:etcd` (V-242445).
- Files have restrictive file permissions set (V-242459).

## Kubernetes Worker Nodes

Kyma uses [Garden Linux](#) as its node operating system. Garden Linux is a Linux distribution with a minimal set of applications optimized for use in Gardener landscapes. Kubernetes worker nodes use a `kubectl` version higher than 1.12.9 (V-242396).

## kubelet

The Kubernetes kubelet fulfills the following DISA STIGs requirements:

- The read-only port flag (V-242387) and anonymous authentication (V-242391) are disabled.
- Explicit authorization is required (V-242392).
- Hostname override is denied (V-242404).
- An SSL certificate authority is configured (V-242420).
- TLS is enabled to enforce TLS encrypted communication sessions (V-242424, V-242425).
- Kernel protection is set (V-242434).
- A connection timeout is set (V-245541).
- No staticPodPath is configured (V-242397).
- The Kubernetes kubelet configuration file (V-242406), certificate authority (V-242450), and config (V-242453, V-242457) are owned by root.
- The Kubernetes kubelet configuration files (V-242407), certificate authority file (V-242449), config file (V-242452, V-242456) have restrictive file permissions set (V-242407).

## kube-proxy

The Kubernetes kube-proxy fulfills the following DISA STIGs requirements:

- A kubeconfig file is not used (V-242447, V-242448).

# Network Separation

## Namespaces

Kyma components are deployed into the `kyma-system` and the `istio-system` namespaces. The default namespace is not used.

## Network Policies

You, the customer, have full access to the cluster and can establish appropriate network policies to restrict network traffic. Because Istio is a default Kyma module, the customer can also establish PeerAuthentication policies to control the network traffic.

## Encryption

All communication in the Kyma cluster is encrypted using TLS 1.2.

# Authentication and Authorization

## Authentication at the API Server

By default, a SAP BTP, Kyma runtime cluster is deployed with a shared tenant of SAP Cloud Identity Services provided by SAP. This Identity Provider (IDP) uses OIDC tokens to authenticate and is configured to enforce multi-factor authentication.

As a best practice, you, the customer, can change the IdP to one controlled by you. This gives you control over the identity lifecycle of your accounts. Learn how to [Configure a Custom Identity Provider for Kyma \[page 3134\]](#).

The Kubernetes API Server fulfills the following DISA STIGs requirements:

- Anonymous authentication is disabled (V-242390).
- Basic authentication is disabled to protect information in transit (V-245542).
- Token authentication is disabled to protect information in transit (V-245543).

## Role-based Access Control

RBAC is enabled on Kyma clusters. The Roles and ClusterRoles deployed by Kyma follow the principle of least privilege.

Following DISA STIGs requirements, the Kubernetes API server enables RBAC as the authorization mode (V-242382).

## High Availability

Kyma worker nodes are distributed over several availability zones in order to prevent outages of the cloud providers' availability zones.

## Logging

### Kubernetes-Native Audit Logging Configuration

Audit Logging of the Kubernetes API server is active, and the logs are written to SAP Platform Logging Service for SAP BTP.

The Kubernetes API server fulfills the following DISA STIGs requirements:

- An audit policy (V-242401), an audit log path (V-242402, V-242465), and audit log retention (V-242464) are set.
- Audit records that provide information about the type of event that has occurred, the source of the event, the results of the event, any users and containers associated with the event are generated (V-242403).
- Audit logs are enabled (V-242461).
- Audit log maximum size (V-242462), and maximum backup are configured (V-242463).

### Container Logging

Kyma modules ship logs that contain information relevant to the containers.

## Related Information

[Security Vulnerability Management in the Kyma Environment \[page 3125\]](#)

[Distributed Denial-of-Service Protection in Kyma \[page 3125\]](#)

[Secure Administration and Operations in the Kyma Environment \[page 3132\]](#)

[Secure Development in the Kyma Environment \[page 3126\]](#)

## 7.6.1.1 Security Vulnerability Management in the Kyma Environment

SAP employs a vulnerability management process to ensure the security of SAP BTP, Kyma runtime. Within this process, we identify, assess, prioritize, remedy, and monitor vulnerabilities. We also provide hotfixes between regular releases for critical vulnerabilities that our security organization identifies.

SAP regularly scans SAP-owned code and the container images that are part of SAP BTP, Kyma runtime, Kyma Control Plane, and Kyma dashboard for known and potential vulnerabilities. The following security scanning technologies are used:

- Static application security testing (SAST) for the proprietary code
- Open source vulnerability management (OSVM) for open-source software that is part of Kyma runtime

### ⓘ Note

SAP runs security scans for Kyma runtime, and you, as a customer, are responsible for scanning your own workloads added to the Kyma cluster. For details, see [Operating Model in the Kyma Environment. \[page 3160\]](#)

The scanning results are constantly monitored and addressed. Vulnerabilities are handled with software updates. If an update is not available, the vulnerability is subject to further analysis to identify its associated risk and implement appropriate measures. For critical vulnerabilities identified by SAP's security organization, hotfixes are applied between regular releases.

### → Remember

Kyma development teams analyze each vulnerability to assess its actual severity in the Kyma environment. Therefore, the severity score may be lowered or identified as false positive.

## 7.6.1.2 Distributed Denial-of-Service Protection in Kyma

A distributed denial-of-service (DDoS) attack is a malicious attempt to disrupt the normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of traffic.

With Kyma comes the default DDoS protection offered by the IaaS provider.

Default DDoS Protection

IaaS Provider

Default DDoS Protection

---

Microsoft Azure

Azure DDoS Protection Basic service

---

IaaS Provider	Default DDoS Protection
Amazon Web Services	AWS Shield Standard
Google Cloud	Standard network DDoS protection

#### → Remember

Because these are cloud providers' default offerings, they are subject to change as per the cloud providers' terms and conditions.

## Web Application Firewalls

It is not possible to configure or enable web application firewalls in SAP BTP, Kyma runtime.

## 7.6.2 Secure Development in the Kyma Environment

Secure development focuses on Pod security, network traffic restriction, Istio sidecar proxy injection, and workload exposure. Learn about the security measures you can take to improve the security of your Kyma environment .

### Pod Security

To learn how to secure your Kubernetes Pods in Kyma, see [Kubernetes Pod Security Recommendations \[page 3127\]](#).

### Restrict and Secure Network Traffic

- Set up Kubernetes [Network Policies](#) to restrict the network traffic between Pods in your namespaces.
- [Enabling Istio Sidecar Proxy Injection \[page 1871\]](#) for your namespaces.

### Expose Workloads Securely

To securely expose workloads in SAP BTP, Kyma runtime, use the Istio and API Gateway modules. As a prerequisite, make sure that you have those two modules added to your Kyma cluster. Then, perform the following tasks:

- [Set Up a Custom Domain for a Workload](#)

- Disable the default gateway. See [Disable or Enable Kyma Gateway](#).
- Set up your own API gateway with one of the following options:
  - [Set Up a TLS Gateway in Simple Mode](#)
  - [Set Up an mTLS Gateway and Expose Workloads Behind It](#)
- Create an [APIRule Custom Resource](#) to securely expose your workloads
  - [Expose and Secure a Workload with JWT](#)
  - [Expose and Secure a Workload with Istio](#)
  - [Expose and Secure a Workload with a Certificate](#)

## Module-Specific Security Considerations

### Serverless Module

See [Security Considerations](#) for the Serverless module.

### Related Information

[Kyma Security Concepts \[page 3120\]](#)

[Secure Administration and Operations in the Kyma Environment \[page 3132\]](#)

[Kubernetes Pod Security Recommendations \[page 3127\]](#)

[Function Security \[page 3131\]](#)

## 7.6.2.1 Kubernetes Pod Security Recommendations

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes. This document shows how to use Pods securely and ultimately have a secure workload. The following steps guide you to use Pods securely and ultimately have a secure workload, depending on your needs.

### 1. Decide on the Basic Security Settings

#### Without Special Requirements: Highest Security

If you are not aware of any special requirements, set up your workload in the most secure way:

- Specify a user for the Pod and container. If your application requires a specific user, change values for the `runAsUser` and `runAsGroup` parameters.
- Run an unprivileged workload.
- Drop all capabilities.
- Use seccomp. For more information, see the [Restrict a Container's Syscalls with seccomp](#).

- Run with a read-only root file system.

Use the following `securityContext` configuration for the Pod and container.

#### Sample Code

```
# Pod spec
  securityContext:
    runAsUser: 10001
    runAsGroup: 30001
    runAsNonRoot: true
    seccompProfile:
      type: RuntimeDefault
# Container spec
  securityContext:
    privileged: false
    allowPrivilegeEscalation: false
    runAsNonRoot: true
    capabilities:
      drop:
        - ALL
    readOnlyRootFilesystem: true
```

### If Your Container Must Run as User Root

If your container must run as user root, use the following setup:

- Do not use the `runAsUser` and `runAsGroup` parameters in the spec or set them to 0.
- Do not use the `runAsNonRoot` parameter.
- Run an unprivileged workload and do not allow privilege escalation.
- Drop all capabilities.
- Use seccomp.
- Run with a read-only root file system.

Use the following `securityContext` configuration for the Pod and container.

#### Sample Code

```
# Pod spec
  securityContext:
    seccompProfile:
      type: RuntimeDefault
# Container spec
  securityContext:
    privileged: false
    allowPrivilegeEscalation: false
    readOnlyRootFilesystem: true
    capabilities:
      drop:
        - ALL
```

## 2. Enable Additional Capabilities

### Add Storage to Your Pod

If you need storage, follow these recommendations:

- Always use volumes instead of local storage in the container.
- Avoid using `hostPath` volumes. If you must use them, follow these rules:
  - **Whenever possible** mount `hostPath` volumes in the read-only mode.
  - **Never** mount the following sensitive paths:
    - `/proc` - a pseudo-file system that provides an interface to kernel data structures
    - `/` - file system's root
    - `/etc` - the directory that usually contains all system-related configuration files
    - `/root` - the home directory of the root user
    - `/var/run/docker.sock` - the Unix socket used to communicate with the Docker daemon
    - `/var/run/crio/crio.sock` - the Unix socket used to communicate with the CRI-O Container Engine
    - `/run/containerd/containerd.sock` - the Unix socket used to communicate with the Containerd container runtime
    - `/home/admin` - the home directory of the admin user
    - `/var/lib/kubelet` - the directory for kubelet-related configuration
    - `/var/lib/kubelet/pki` - the directory containing the certificate and private key of the kubelet
    - `/etc/kubernetes` - the directory containing Kubernetes-related configuration
    - `/etc/kubernetes/manifests`
  - Use only the `File` or `Directory` `hostPath.type`. Usually use `File`, and `Directory` as a fallback.
  - Some files and directories on the host can only be accessible by root. Read more in the [Kubernetes documentation](#).

Consider the following template for mounting `hostPath` volumes in order to create Kubernetes manifests:

#### ↔ Sample Code

```
# Pod spec
volumes:
  - name: <name>
    hostPath:
      path: <file or directory on the node>
      type: <File> or <Directory>
# Container spec
volumeMounts:
  - mountPath: <path where hostpath has to be mounted>
    name: <name>
    readOnly: true
```

- Use `emptyDir` volumes for temporary file systems.

If your application needs a temporary file system, for example, for caching purposes, use `emptyDir` volumes **instead of** mounting `hostPath` or making the root file system writable. Also, remember to always define a size limit for the volume. Use the following template:

#### ↔ Sample Code

```
# Pod spec
```

```
volumes:
- name: <name>
  emptyDir:
    sizeLimit: <size of the volume>
# Container spec
volumeMounts:
- mountPath: <path where the file system has to be mounted>
  name: <name>
```

## Add Other Capabilities

- Before you add new capabilities to your Pod or container configuration, **always drop all** capabilities. Each container runs with a fixed set of default capabilities. You can find the containerd v1.7 default capabilities in [this code](#).
- Add only the capabilities you really need.
- If possible, avoid the following capabilities that can be exploited to escalate privileges:
  - CAP\_CHOWN
  - CAP\_DAC\_OVERRIDE
  - CAP\_DAC\_READ\_SEARCH
  - CAP\_SETUID, CAP\_SETGID
  - CAP\_NET\_RAW
  - CAP\_SYS\_ADMIN
  - CAP\_SYS\_PTRACE
  - CAP\_SYS\_MODULE
  - CAP\_FORMER
  - CAP\_SETFCAP

If your application doesn't work as expected, you may need some additional capabilities. Find the full list of the available capabilities on the [Linux manual page](#). Check which capabilities are missing using the [Inspector Gadget](#) tools. You can [install the toolset as a kubectl plugin using krew](#). After it's installed, use the `kubectl gadget trace capabilities` command.

## 3. Consider Special Features

### Accessing Parts of the /proc File System

The `proc` file system provides a lot of critical information about the system and its running processes. To learn more, read [Exploring the Linux /proc file system](#) and [Important Linux /proc file system files you need to know](#). By default, container runtimes mask certain parts of the `/proc` file system from inside a container in order to prevent potential security issues.

#### ⚠ Caution

You should change settings around the `/proc` file system only if you are aware of the impact. Changing the configuration may expose your system to security issues.

- Check if you really need to use the `/proc` file system. For example, if you want to use it for image building purposes, check the latest version of your build tool because many of them no longer need it.

- If you really need to use the `/proc` file system, only do so for a nested container. Never expose the `/proc` file system of your host system to a container.
- If your application needs access to parts of the `/proc` file system, you can set the `procMount` field in the `PodSpec` to `unmasked` to remove the restriction on the visibility of the `/proc` file system from inside the container.

## Seccomp

Seccomp is a Linux kernel feature that allows filtering system calls made by a process. It is used to restrict the set of system calls that a process can make.

If, in `SecurityContext`, the `seccompProfile` is set to `RuntimeDefault`, the container uses the default seccomp profile provided by the container runtime.

Read more to learn about:

- [Allowed system calls](#) in the default profile of containerd.
- The full list of [Linux system calls](#).
- How to work with Seccomp profiles in Kubernetes - [Restrict a Container's Syscalls with seccomp](#).
- How to set up a [Custom Seccomp Profile](#) in Gardener.

## Host Namespaces

Host namespaces (Process ID namespace, Inter-Process Communication namespace, and network namespace) allow you to access shared information. You can also use host namespaces to elevate privileges. By removing namespaces from Pods, you reduce isolation and circumvent the protection models that containers are based on. It allows the processes in the Pod to perform tasks as if they were running natively on the host.

In the Pod spec, the following fields represent the host namespaces:

- `hostPID`: Process ID namespace
- `hostIPC`: Inter-Process Communication namespace
- `hostNetwork`: network namespace

The three fields are by default set to `false` and should remain so.

Usually, there is no need to remove this isolation and therefore, Pods should not have access to host namespaces.

### ⚠ Caution

Removing namespaces from Pods is a very advanced feature. Use it only if you are certain you need it, for example, for low-level observation of other containers.

## 7.6.2.2 Function Security

When creating Functions, make sure you understand how they work to avoid potential threats.

To eliminate potential security risks when using Functions, bear in mind these few facts:

- By default, JSON Web Tokens (JWTs) issued by an OIDC-compliant identity provider do not provide the `scope` parameter for Functions. This means that if you expose your Function and secure it with a JWT,

you can use the token to validate access to all Functions within the cluster as well as other JWT-protected services.

- There are no authorization policies defined that would restrict Functions' access to other resources within the Namespace. If you deploy a Function in a given Namespace, it can freely access all events and APIs of services within this Namespace.
- All administrators and regular users who have access to a specific Namespace in a cluster can also access:
  - Source code of all Functions within this Namespace
  - Internal Docker registry that contains Function images

## 7.6.3 Secure Administration and Operations in the Kyma Environment

Secure administration and operations focus on landscape setup, authentication, authorization, and Istio access logs. Learn about the security measures you can take to improve the security of your Kyma environment.

### Landscape Setup

While creating a staged development environment is a good idea in any case, there are some considerations specific to Kyma you might want to take into account. For more information, see [Staged Development with the Kyma Environment](#).

### Authentication

SAP BTP, Kyma runtime uses OpenID Connect for authentication. Kyma runtime is configured to use a default shared SAP Cloud Identity Services tenant, SAP ID service, as an identity provider. This is a good starting point for development and testing purposes. However, for production scenarios, it is recommended to get your own SAP Cloud Identity Services tenant. For more information, see [Authentication in the Kyma Environment \[page 3133\]](#).

### Authorization

Kyma uses Kubernetes Role-Based Access Control (RBAC) and assures during provisioning that a user who creates and owns a particular runtime is given the cluster-admin role. Users with the cluster-admin role can define any additional cluster roles or use those defined in Kyma and bind them to other users from Kyma dashboard or with the kubectl CLI tool. See [Assign Roles in the Kyma Environment \[page 3137\]](#). For recommendations on setting up roles and permissions in Kyma, see [Role-Based Access Control \(RBAC\) in Kyma](#).

### Note

For more information about user and member management in the SAP BTP cockpit, see [User and Member Management \[page 104\]](#).

## Istio Access Logs

Collecting Istio access logs can help indicate the core concepts of monitoring (latency, traffic, errors, and saturation) and capture critical aspects of system behavior. By [Configuring Istio Access Logs \[page 1883\]](#) you monitor and analyze the traffic in your Kyma cluster. Then, send the logs to a central platform, for example, a Security Information and Event Management (SIEM) system, for threat detection and incident response.

## Related Information

[Kyma Security Concepts \[page 3120\]](#)

[Secure Development in the Kyma Environment \[page 3126\]](#)

[Authentication in the Kyma Environment \[page 3133\]](#)

[Configure a Custom Identity Provider for Kyma \[page 3134\]](#)

[Assign Roles in the Kyma Environment \[page 3137\]](#)

[Auditing and Logging Information in Kyma \[page 3139\]](#)

[Deprecation Trial for Google's Third-Party Cookies \[page 3142\]](#)

### 7.6.3.1 Authentication in the Kyma Environment

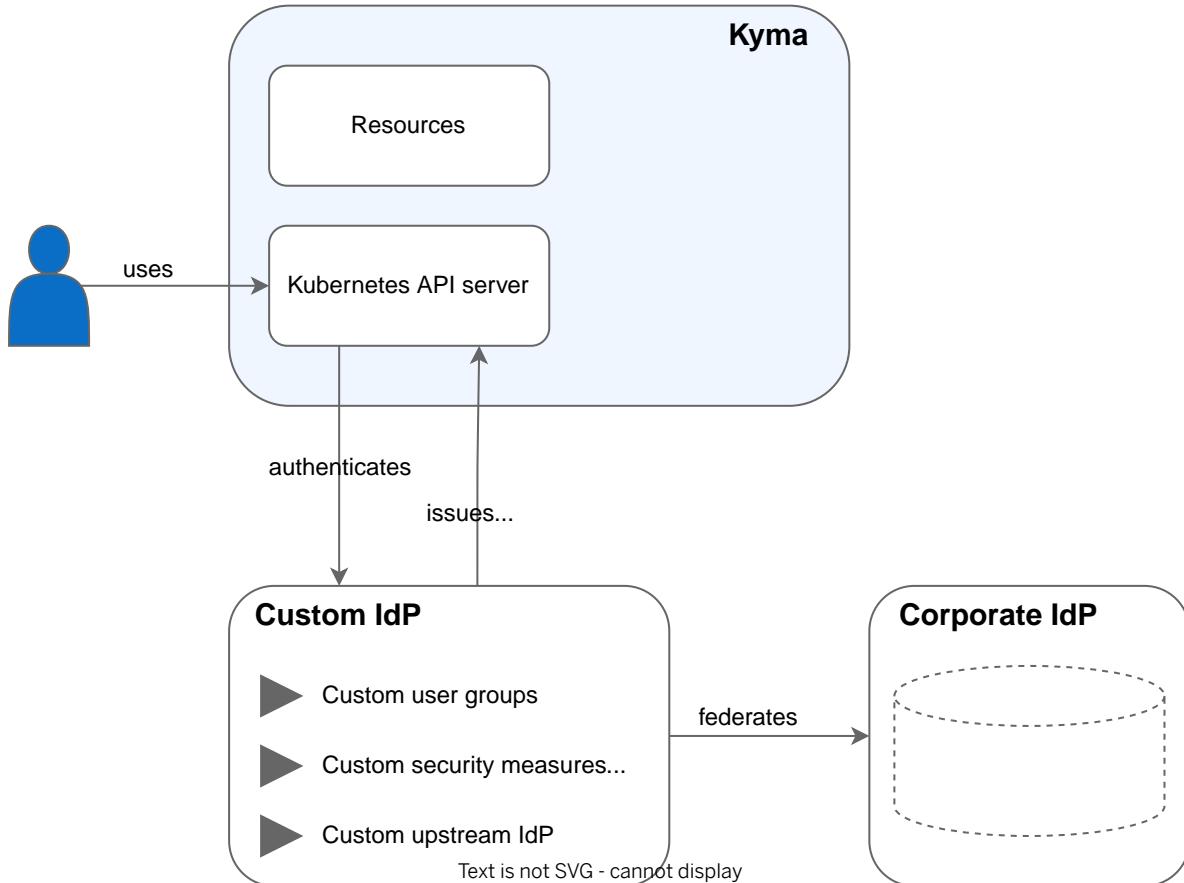
To authenticate in the Kyma environment, you can either use the default identity provider (IdP) or set up a custom identity provider.

## Default vs. Custom Identity Provider

SAP BTP, Kyma runtime uses OpenID Connect for authentication. Kyma runtime is configured with a default IdP: a shared [SAP Cloud Identity Services](#) tenant. This is a good starting point for development and testing purposes. For production scenarios, it's recommended that you set up your own SAP Cloud Identity Services tenant which provides the following features:

- manages users
- manages user groups and binds cluster roles to the user groups instead of individual usernames in your Kyma runtime
- configures your own risk-based security measures, such as two-factor authentication, IP ranges, etc.

- delegates authentication to an upstream corporate IdP to prevent introducing redundant identities and to benefit from SSO, based on identity provided "centrally"



Learn how to [Get Your Tenant](#) from SAP Cloud Identity Services.

Alternatively, you can choose any other service compliant with OpenID Connect.

Having an SAP Cloud Identity Services tenant, you can configure it as the custom identity provider of your Kyma cluster. See [Configure a Custom Identity Provider for Kyma \[page 3134\]](#).

### 7.6.3.1.1 Configure a Custom Identity Provider for Kyma

Enable the Kyma environment with a custom identity provider (IdP).

#### Prerequisites

- Install [kubectl oidc-login](#). This is also needed when you log in to Kyma dashboard, as the dashboard uses the kubeconfig file, and it requires the kubectl oidc-login tool to work.
- If you choose to use SAP Cloud Identity Services as custom IdP, you have configured your tenant as OICD provider for your Kyma cluster. For details, see [Configure OpenID Connect Application for Authorization Code Flow](#).

- Configure the Kyma dashboard URL (`https://dashboard.kyma.cloud.sap`) and the localhost for kubectl authentication (`http://localhost:8000`) as allowed callback URLs at your IdP provider, so that authenticated users can be redirected back to the Kyma application. See [Redirect URLs, Post Logout Redirect URI Rules](#).

→ Tip

It is recommended to use SAP Cloud Identity Services tenant as a custom IdP. It allows you to use SAP Cloud Identity Services as a proxy to integrate your corporate identity provider. See [Get Your Tenant](#). You could also configure SAP Cloud Identity Services to use a third-party IdP if you already have one.

## Context

When you create a new Kyma instance in the SAP BTP cockpit from the Service Marketplace, you can configure your custom OpenID Connect IdP to authenticate users in your Kyma runtime.

If you've already created your Kyma environment, you can also apply the custom IdP configuration and set up administrators during your Kyma instance update operation by providing the details as an array of strings in the respective fields.

## Procedure

- Go to ► [Services](#) ► [Service Marketplace](#) and [Create a Kyma Instance \[page 2992\]](#).

- In the [Parameters](#) view, fill in the following information:

- issuerURL* - the URL of the OpenID issuer (use the `https` schema)
- clientID* - the client ID for the OpenID client
- usernameClaim* - the name of a custom OpenID Connect claim for specifying a username
- groupsClaim* - the name of a custom OpenID Connect claim for specifying user groups

ⓘ Note

An identity provider token consists of fields (claims) that are included in it. For example:

```
{
  sub: "john.doe@sap.com"
  groups: ["SAP fans", "Identity Provider", "OpenID"]
}
```

When you configure `usernameClaim`, Kubernetes knows which field (claim) includes the user represented by a given token.

Similarly, when you configure `groupsClaim`, Kubernetes knows to which groups the user is assigned.

- signingAlgs* - the list of allowed cryptographic algorithms used for token signing. The allowed values are defined by [RFC 7518](#).

- *usernamePrefix* - the prefix for all usernames. If you don't provide it, username claims other than "email" are prefixed by the *issuerURL* to avoid clashes. To skip any prefixing, provide the value as `-`.
- *administrators* - the list of usernames meant to be administrators. Users identified with those usernames get the role `cluster-admin` during provisioning of your Kyma runtime.

You can also provide the configuration as a JSON object:

```
"oidc": {
    "issuerURL": "{issuerURL}",
    "clientID": "{clientID}",
    "usernameClaim": "sub",
    "groupsClaim": "groups",
    "signingAlgs": ["RS256"],
    "usernamePrefix": "-"
},
"administrators": [
    "example_1@mail.com",
    "example_2@mail.com"
]
```

### ⓘ Note

If you want to revert to the default settings, use the following configuration:

```
"oidc": {
    "issuerURL": "https://kyma.accounts.ondemand.com",
    "clientID": "12b13a26-d993-4d0c-aa08-5f5852bbdff6",
    "groupsClaim": "groups",
    "signingAlgs": ["RS256"],
    "usernamePrefix": "-",
    "usernameClaim": "sub"
},
"administrators": [
    "example_3@mail.com",
    "example_4@mail.com"
]
```

The email addresses must be recognised by `https://kyma.accounts.ondemand.com`.

3. Select [Create](#).

## Results

Your Kyma environment is instantiated with a custom IdP.

## Related Information

[Authentication in the Kyma Environment \[page 3133\]](#)

## 7.6.3.2 Assign Roles in the Kyma Environment

Kyma uses roles to manage access within the cluster, which give the assigned users the permissions suitable for their purposes.

### Prerequisites

The `cluster-admin` role is assigned to your account.

#### ⓘ Note

After creating a Kyma cluster, you become an admin of this instance and the `cluster-admin` role is assigned to you by default. As the `cluster-admin`, you can assign roles to other users. Only the administrator who created the current subaccount can use the SAP BTP cockpit to assign the `cluster-admin` role to other members.

### Context

You can assign roles to a group of users only if you use a custom identity provider. Assigning roles in Kyma is based on the [Kubernetes role-based access control \(RBAC\)](#). You can see the list of available roles in Kyma dashboard when you create a Role Binding. Once you become the admin, your user name is read from the SAP BTP (RBAC) concept and is passed to the Kyma provisioner to be bound to the `cluster-admin` role in Kyma.

### Procedure

1. Log in to Kyma dashboard. The URL is in the [Overview](#) section of your subaccount.
2. In Kyma dashboard, select or create the namespace in which you want to assign roles.
3. To create a role binding, go to + [Create Role Binding](#). Fill in the required fields:
  - a. As [Name](#), insert the name of your binding.
  - b. As [Role Type](#), choose [ClusterRole](#).
  - c. As [Role](#), choose the required role from the list.
  - d. As [Kind](#), choose [User](#).
  - e. As [User name](#), insert the user email address.
4. Select [Create](#).

## Results

The users have the required permissions within the specified namespace. If the users don't have additional Cluster Role Binding to list the namespaces, they can still access the Kyma dashboard overview but must enter the required namespace name manually.

## Next Steps

If the permissions of the default role aren't sufficient, clone the role and add the missing resources.

## Related Information

[Role-Based Access Control \(RBAC\) in Kyma](#)

### 7.6.3.2.1 Overwrite Kyma Administrators

In some cases, you may need to overwrite usernames of the current administrators, for example, if they forget their password or leave the organization and nobody can access your Kyma runtime.

#### Prerequisites

- You're the subaccount administrator.

#### Context

To overwrite the names for the `cluster-admin` role, update the Kyma instance and provide a new value for the [`administrators`](#) field.

##### ⚠ Caution

This procedure overwrites current administrators and shouldn't be used to add new ones. Treat it as an emergency self-service procedure in case of lost access. To provide access to new users, the `cluster-admin` should create a [RoleBinding](#) and/or a [ClusterRoleBinding](#) in the runtime. See [RoleBinding and ClusterRoleBinding](#) in the official Kubernetes documentation.

## Procedure

1. In the SAP BTP cockpit, go to your Kyma instance and select *Update*.
2. Provide a new set of administrators' usernames as an array of strings.

### → Remember

Use administrators' email addresses as their usernames.

In a JSON file, use the following structure:

```
"administrators": [  
    "example_1@mail.com",  
    "example_2@mail.com",  
    "example_3@mail.com"  
]
```

3. Click *Update*.

## Results

Your Kyma instance is updated with the new administrators' usernames.

## Related Information

[Provisioning and Updating Parameters in the Kyma Environment \[page 3002\]](#)

### 7.6.3.3 Auditing and Logging Information in Kyma

Kyma runtime collects audit and application logs.

## Audit Logs

Audit logs are records that provide evidence of events, actions, or operations. The audit log records the following data:

- Security and system events that include administrative activities, potential threats, and evidence in the case of a breach.
- Changes introduced to the system, applications, and components.

Kyma runtime collects audit logs for configuration changes to the runtime setup itself and Kubernetes resources managed by the Kubernetes API server of the runtime. Usually, those logs contain the user account

(technical account or email address) and the client IP address of the subject who triggered the changes. No other personal data is stored in the audit logs. On average, the logs store the data for 90 days. If you want to store any other personal data, be cautious and bear in mind the recommendations provided in [Data Protection and Privacy \[page 3109\]](#).

## Retrieving Audit Log Entries

As a user, you don't have direct access to the audit logs. If you want to get information stored in the audit log system, open a [support ticket](#).

## Events Written in Audit Logs

Here you can find a list of the security events that are logged by SAP BTP, Kyma runtime.

Events Written in Audit Logs

Event grouping	What events are logged	Additional information
Secrets, ConfigMaps, and TokenReviews	CREATE, UPDATE, PATCH, and DELETE operations on the metadata level	The payload is not logged. Resources: <ul style="list-style-type: none"><li>"secrets"</li><li>"configmaps"</li><li>"tokenreviews"</li></ul>

Event grouping	What events are logged	Additional information
All standard Kubernetes resources in core and extensions	CREATE, UPDATE, PATCH, and DELETE operations on the request level	<p>Resource groups:</p> <ul style="list-style-type: none"> <li>• "" # core</li> <li>• "admissionregistration.k8s.io"</li> <li>• "apiextensions.k8s.io"</li> <li>• "apiregistration.k8s.io"</li> <li>• "apps"</li> <li>• "authentication.k8s.io"</li> <li>• "authorization.k8s.io"</li> <li>• "autoscaling"</li> <li>• "batch"</li> <li>• "certificates.k8s.io"</li> <li>• "coordination.k8s.io"</li> <li>• "extensions"</li> <li>• "metrics.k8s.io"</li> <li>• "networking.k8s.io"</li> <li>• "policy"</li> <li>• "rbac.authorization.k8s.io"</li> <li>• "scheduling.k8s.io"</li> <li>• "settings.k8s.io"</li> <li>• "storage.k8s.io"</li> </ul>
All Kyma resources	CREATE, UPDATE, PATCH, and DELETE operations on the request level	<p>Resource groups:</p> <ul style="list-style-type: none"> <li>• "applicationconnector.kymaproject.io"</li> <li>• "compass.kymaproject.io"</li> <li>• "eventing.kymaproject.io"</li> <li>• "gateway.kymaproject.io"</li> <li>• "serverless.kymaproject.io"</li> <li>• "telemetry.kymaproject.io"</li> <li>• "operator.kymaproject.io"</li> </ul>

Event grouping	What events are logged	Additional information
All resources of Kyma dependencies	CREATE, UPDATE, PATCH, and DELETE operations on the request level	Resource groups: <ul style="list-style-type: none"> <li>• "cert.gardener.cloud"</li> <li>• "ory.sh"</li> <li>• "services.cloud.sap.com"</li> <li>• "extensions.istio.io"</li> <li>• "install.istio.io"</li> <li>• "networking.istio.io"</li> <li>• "security.istio.io"</li> <li>• "telemetry.istio.io"</li> </ul>

## Application Logs of System Components

Kyma runtime collects application logs provided by the Kyma system components. Those logs reflect regular events that occurred in Kyma runtime and must not include any personal or sensitive data. These are typically system-related activities and operations you can analyze to get more information about performance or to inspect potential problems.

### Related Information

<https://kubernetes.io/docs/tasks/debug/debug-cluster/audit/> ↗

[Audit Logging in the Cloud Foundry Environment](#)

[Audit Logging in the Neo Environment](#)

[Access Kyma Application Logs \[page 3014\]](#)

### 7.6.3.4 Deprecation Trial for Google's Third-Party Cookies

Google has announced its plan to phase out support for third-party cookies by the third quarter of 2024. See [The Privacy Sandbox Timeline for the Web](#) ↗ and [Prepare for Third-Party Cookie Restrictions](#) ↗.

Third-party cookies are cookies that are set by domains other than the one a user is directly visiting. They are commonly used for online advertising and tracking user behavior across multiple websites. They can be set through JavaScript, iframe, embedded images, or media. SAP primarily uses third-party cookies for service management and single sign-on.

Google offers the [third-party deprecation trial](#) ↗ that allows re-enabling third-party cookies for a page embedded within a cross-site iframe until December 27, 2024. To enable the depreciation trial, all applications' responses must include a depreciation trial token obtained for each affected domain. At SAP, the process of obtaining the tokens is handled centrally.

There are three different ways to add the deprecation trial token - you can either provide it in an HTTP header, use a meta tag, or inject the token with JavaScript. For SAP BTP, Kyma runtime workloads, it is recommended that you use the HTTP header. You must add the token manually for each of the affected workloads.

## Technical Details

In SAP scenarios, the first request to the origin in a session typically requires cross-site cookies. Sending the `Critical-Origin-Trial` header in the HTTP response with the trial name `Tpcd` causes the browser to retry the request with third-party cookies enabled.

For SAP BTP, Kyma runtime, it is not possible to identify the initial request, so the `Critical-Origin-Trial` header is included in every response. This approach has been accepted by Google.

Each HTTP response must also include a valid depreciation trial token for the used domain in the following HTTP header:

```
Origin-Trial: <token-domain>
```

For domains specific to a particular customer, use the token(s) provided by that customer. For the domain `ondemand.com` managed by SAP, use the following token:

```
Avu6rn7emV5gK8gvyGH1X8TMqM9uo1FacP2j/RWTq+8j+yKnqcT00TQh0bXJ/7QntxD4/  
JzXv8aXoqxxZQuqXgYAAABdeyJvcmlnaW4iOiJodHRwczovL29uzGVtYW5kLmNvbTo0NDMiLCJmZWF0dx  
J1IjoiVHBjZCIsImV4cGlyeSI6MTczNTM0Mzk5OSviaXNTdWJkb21haW4iOnRydWV9
```

The token consists of the feature name (`Tpcd`) and its source, along with other attributes. If an application or service can be accessed from multiple domains simultaneously, you can add one token per domain as a comma-separated list.

```
Origin-Trial: <token-domain1>, <token-domain2>, ...>
```

## Related Information

[Apply Third-Party Cookie Deprecation Trial Tokens for Kyma Runtime Workloads \[page 3144\]](#)

[Preparing and Testing Your Solution for Third-Party Cookie Deprecation](#)

## 7.6.3.4.1 Apply Third-Party Cookie Deprecation Trial Tokens for Kyma Runtime Workloads

Learn how to use EnvoyFilter or Istio VirtualService to add a third-party cookie deprecation token to an HTTP response for a particular workload exposed under the domain `ondemand.com`.

### Prerequisites

- You have the Istio module added.
- You have a workload deployed.

### Add a Third-Party Cookie Deprecation Trial Token Using EnvoyFilter

#### Context

To modify the responses for a particular workload within the Istio service mesh, you can use EnvoyFilter with an additional HTTP filter added to the Envoy proxy configuration of the workload's sidecar. To do this, use the HTTP Lua filter.

#### Procedure

1. In the following EnvoyFilter's configuration, replace the placeholders.

Option	Description
<code>&lt;workload-tpcd-ef-name&gt;</code>	The name of your EnvoyFilter.
<code>&lt;workload-namespace&gt;</code>	The namespace with the workload for which you apply the third-party cookie deprecation token.
<code>&lt;workload-label-selector&gt;</code>	The label selector for the workload.

```
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: <workload-tpcd-ef-name>
  namespace: <workload-namespace>
spec:
  workloadSelector:
    labels:
      <workload-label-selector>
  configPatches:
  - applyTo: HTTP_FILTER
    match:
```

```

    listener:
      filterChain:
        filter:
          name: "envoy.filters.network.http_connection_manager"
          subFilter:
            name: "envoy.filters.http.router"
    patch:
      operation: INSERT_BEFORE
      value:
        name: envoy.lua
        typed_config:
          "@type": "type.googleapis.com/
envoy.extensions.filters.http.lua.v3.Lua"
          inlineCode: |
            function envoy_on_response(response_handle)
              response_handle:headers():add("Origin-Trial",
"Avu6rn7emV5gK8gvyGH1X8TMqM9uo1FacP2j/RWTq+8j+yKnqcTO0TQh0bxJ/7QntxD4/
JzXv8aXoqxxZQuqXgYAAABdeyJvcmlnaW4iOjodHRwczovL29uZGVtYW5kLmNvbTo0NDMiLCJmZWF
0dxJ1IjoiVHBjZCIsImV4cGlyeSI6MTczNTM0Mzk5OSwiaXNTdWJkb21haW4iOnRydWV9")
              response_handle:headers():add("Critical-Origin-Trial", "Tpcd")
            end

```

## 2. Apply the EnvoyFilter configuration.

Once the EnvoyFilter is applied, any request that includes the specified token will always have the necessary HTTP response headers added.

## Results

Responses from the workload endpoint include the specified HTTP headers. See an example response:

```

HTTP/2 200
server: istio-envoy
date: Thu, 11 Apr 2024 13:17:11 GMT
content-type: application/json
content-length: 599
access-control-allow-origin: *
access-control-allow-credentials: true
x-envoy-upstream-service-time: 3
origin-trial: Avu6rn7emV5gK8gvyGH1X8TMqM9uo1FacP2j/RWTq+8j+yKnqcTO0TQh0bxJ/
7QntxD4/
JzXv8aXoqxxZQuqXgYAAABdeyJvcmlnaW4iOjodHRwczovL29uZGVtYW5kLmNvbTo0NDMiLCJmZWF
0dx
J1IjoiVHBjZCIsImV4cGlyeSI6MTczNTM0Mzk5OSwiaXNTdWJkb21haW4iOnRydWV9
critical-origin-trial: Tpcd
{
  "headers": {
    "Accept": "*/*",
    "Host": "httpbin.xxx.ondemand.com",
    "User-Agent": "curl/8.4.0",
    "X-Envoy-Attempt-Count": "1",
    "X-Envoy-Expected-Rq-Timeout-Ms": "180000",
    "X-Envoy-Internal": "true",
    "X-Forwarded-Client-Cert": "...",
    "X-Forwarded-Host": "httpbin.xxx.ondemand.com"
  }
}

```

# Add a Third-Party Cookie Deprecation Trial Token Using Istio VirtualService

## Context

If you use Istio VirtualServices rather than APIRules to expose Kyma workloads, adapt your Istio VirtualService configuration to include the third-party cookie deprecation token in HTTP response headers.

## Procedure

- In the following configuration of Istio VirtualService, replace the placeholders.

Option	Description
<workload-with-tpcd-trial-token>	The name of the workload for which you apply a third-party cookie deprecation token.
<workload-namespace>	The namespace of the workload.

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: <workload-with-tpcd-trial-token>
  namespace: <workload-namespace>
spec:
  gateways:
    - kyma-system/kyma-gateway
  hosts:
    - httpbin.ddd.ondemand.com
  http:
    - corsPolicy:
        allowHeaders:
          - Authorization
          - Content-Type
          - '*'
        allowMethods:
          - GET
          - POST
          - PUT
          - DELETE
          - PATCH
        allowOrigins:
          - regex: .*
        headers:
          request:
            set:
              x-forwarded-host: httpbin.ddd.ondemand.com
          response:
            set:
              origin-trial: <Avu6rn7emV5gK8gvyGH1X8TMqM9uo1FacP2j/RWTq+8j+yKnqcTO0TQh0bXJ/7QntxD4/JzXv8aXoqxxZQuqXgYAAABdeyJvcmlnaW4iOjodHRwczovL29uZGVtYW5kLmNvbTo0NDMiLCJmZWFOdXJ1IjoivHbjZCIsImV4cGlyeSI6MTczNTM0Mzk5OSwiaXNTdWJkb21haW4iOnRydWV9>
              critical-origin-trial: Tpcd
      match:
        - method:
            regex: ^(GET)$
```

```
uri:
  regex: /.*
route:
- destination:
  host: httpbin.<workload-namespace>.svc.cluster.local
  port:
    number: 8000
  weight: 100
```

2. Apply the Istio VirtualService configuration.

Once the Istio VirtualService configuration is applied, any request that includes the specified token will always have the necessary HTTP response headers added.

## Results

Responses from the workload endpoint include the specified HTTP headers.

### 7.6.3.5 Verify Image Signatures in the Kyma Environment

Image signing and image signature verification are important countermeasures against security threats. They help you ensure the authenticity and integrity of the application you deploy in the runtime. You can be sure that what you deploy is what you have built in the build pipeline. While image signing happens in the automated Continuous Delivery (CD) workflow, image signature verification takes place at the time the workload is scheduled in the runtime. For this purpose, Kyma uses Warden, which is added to your cluster by default and ensures that the Kyma workloads are authentic.

Warden is automatically enabled in the namespaces managed by Kyma (`kyma-system`, `istio-system`). Therefore, you can be sure the Kyma-managed workloads are authentic. Use the `kubectl get namespaces -l namespaces.warden.kyma-project.io/validate=enabled` command to list the Kyma system namespaces, protected by Warden.

#### ⚠ Caution

When you enable third-party Kubernetes tools that mutate workloads (for example, Dynatrace that adds init containers), you must exclude the Kyma-managed namespaces, which are protected by Warden, from the tool config. Otherwise, Warden blocks unknown images. Mind that disrupting the Kyma-managed namespaces may lead to an inability to uphold the Service Level Agreement (SLA).

## Related Information

[GitHub repository: Warden](#) ↗

# 8 Getting Support

To get assistance, use the available support channels provided by SAP for Me.

## ⓘ Note

This section is applicable for all multicloud environments. To find which environments are available in your region, see [Regions \[page 17\]](#).

## Prerequisites

Before you report a case, ensure the following:

### Check Platform Status

- **Running on the AWS, Azure, or GCP regions:** visit [SAP Trust Center](#). You can check:
  - availability by service on [Cloud Status](#) tab page;
  - availability by region on the [Data Center](#) tab page.
- **Running on the China (Shanghai) region:** check <https://status.cn40.platform.sapcloud.cn/>.
- **Running on the Government Cloud (US) region:** planned downtimes and outage communication are sent through e-mail to the initial administrator of your global account.

### Check Tools Versions

Make sure that you've the latest versions of the tools you're using (recommended), or at least the versions you're using are still supported. For more information, see [SAP Development Tools](#).

## ⓘ Note

Trial users can ask for support by posting a question in [SAP Community](#). However, they should first check for answers in [SAP Community](#) and [Guided Answers](#).

### Try Out Built-In Support Within SAP BTP Cockpit

When using SAP BTP cockpit as an S-user, try Built-In Support to get customer help within the cockpit. See [Built-In Support](#).

### Learn the Best Practices and Offerings

Familiarize with the guidelines how to engage with support. See [Support from SAP Best Practices & Offerings](#).

## Procedure

To report a case in SAP for Me, proceed as follows:

1. Open [SAP for Me](#) and log on.
2. Open the *Services & Support* dashboard.
3. (Optional) You can use the *Knowledge Search* tab for existing solutions in:
  - SAP Notes & KBAs
  - SAP Community
  - SAP Support Portal
  - Product Documentation
  - Guided Answers
4. Use the *Get Support* application within SAP for Me to get assistance via the available support channels.  
For more information about the specifics of each support channel and to learn how to report technical issues, see [Get Support Application](#).  
For any other non-technical assistance, see [Customer Interaction Center](#).

## Creating a Support User

Sometimes you might need to create a support user to troubleshoot issues in your system. For more information about how and when to create a support user, see [3543065 - Best Practices for Creating Support Users in Your BTP System](#).

## Related Information

- [Users and Authorizations for SAP One Support Launchpad](#)
- [Cloud Management Tools — Feature Set Overview \[page 129\]](#)
- [Gather Support Information \[page 3151\]](#)
- [Platform Updates and Notifications \[page 3152\]](#)
- [Cloud Availability Center](#)
- [Road Map for Cloud Foundry Environment](#)
- [Road Map for Kyma Runtime](#)
- [Road Map for ABAP Environment](#)

## 8.1 Providing Details for SAP HANA Service Database Problems

If your problem is related to a SAP HANA service database, the details you need to provide differ depending on the environment or infrastructure provider the database is provisioned in.

Infrastructure Provider	Details You Need to Provide	How to Find the Details You Need
Azure regions	The URL of the space the database is provisioned in	<ol style="list-style-type: none"><li>1. In the SAP BTP cockpit, navigate to the org and space the database is provisioned in.</li></ol>
AWS regions  (databases provisioned before June 4, 2018)		<ol style="list-style-type: none"><li>2. In the navigation area, go to  SAP HANA  Database Systems .</li><li>3. Choose the affected <i>Database System</i>.</li><li>4. Copy the URL from the overview of the affected database system. Example URL: <code>https://account.hana.ondemand.com/cockpit#/globalaccount/&lt;id&gt;/subaccount/&lt;id&gt;/org/&lt;id&gt;/dbsystems/&lt;id&gt;/overview</code></li></ol>
AWS regions  (databases provisioned after June 4, 2018)	The service instance ID	<ol style="list-style-type: none"><li>1. In the SAP BTP cockpit, navigate to the space the SAP HANA service instance is provisioned in.</li><li>2. In the navigation area, choose  Services  Service Instances .</li><li>3. In the list of available services, find your SAP HANA service instance and choose  Open Dashboard from the <i>Actions</i> column.</li><li>4. Copy the service instance GUID under  Details  ID .</li></ol> <p>Example GUID: <code>ad539b83-39bd-4ff4-8b41-1a5dfdfca7ea</code></p>
Google Cloud regions		
China (Shanghai) region	See <a href="#">Getting Support</a>	
Government Cloud (US) region	Please contact your operator.	

## Related Information

[Providing Details for Database Problems in the Neo Environment](#)

## 8.2 Gather Support Information

The Eclipse tools come with a wizard for gathering support information in case you need help with a feature or operation (during deploying/debugging applications, logging, configurations, and so on).

### Context

The wizard collects the information in a ZIP file, which can be later sent to the support team. This way, the support developers can get better understanding of your environment and process the issue faster.

### Procedure

1. From the Eclipse IDE, choose  [Help](#)  [Collect Support Information](#).
2. The launched wizard lists the default components to be collected, depending on the tools you've installed. If you need the support team to look at specific resources, expand the [Additional Data](#) section and select the relevant items.

#### Note

If you select [Screenshot](#), your currently open Eclipse windows and views are snapped as a picture and added to the ZIP file. Make sure you don't reveal sensitive information.

3. In the [File Name](#) field, specify the ZIP file name and location.
4. Choose [Finish](#).

### Next Steps

You can create a support ticket, attach the files to it, and send it to the corresponding support team. For more information, see [Getting Support \[page 3148\]](#).

## 8.3 Platform Updates and Notifications

SAP BTP is a dynamic product, which has continuous production releases (updates). To get notifications for the new features and fixes every release, subscribe at [What's New for SAP Business Technology Platform](#). For more information, see [Subscribing to What's New Notifications](#).

### Reasons for Region Updates

Regions are updated in the following cases:

- **Biweekly updates** (standard) - aligned with the contractual obligations to customers and partners. Such updates usually don't affect productive applications, because most services support zero downtime maintenance. For more information about the biweekly updates, see [Consolidated Release Schedules for SAP BTP](#), [Intelligent Enterprise Suite: Harmonized release calendar for SAP Cloud products](#), and [Service Level Agreement for SAP Cloud Services](#).
- **Immediate updates** - fixes required for bugs that affect productive application operations, or due to urgent security fixes. In some cases, this might lead to downtime or application restart, for which the application groups receive a notification.
- **Major upgrades** - happen rarely, in a bigger maintenance window, up to four times per year. For the time frames of the services' major upgrades, see [Service Level Agreement for SAP Cloud Services](#). We let you know about these upgrades 4 weeks in advance.

### Announcements and Subscriptions [AWS, Azure, or GCP Regions]

You can follow the availability of the platform at [SAP Trust Center](#). You can check:

- the availability by service on the [Cloud Status](#) tab page;
- the availability by region on the [Data Center](#) tab page.

In addition, you can use [Cloud Availability Center](#) (CAC) to check what cloud products and services are assigned to your customer ID for notifications. CAC provides you with a personalized dashboard based on SAP Fiori. It's designed to give you quick access to relevant information about the cloud products that you own and their availability and maintenance status. For more information, see [CAC User Guide](#).

You can find information about the supported  [Cloud Product](#)  [Cloud Service](#) at [Supported Products](#).

The cloud products are tagged with  (orange check mark), while the cloud services with  (gray check mark).

Note: The foundational components of the platform are included in  [SAP Extension Suite – Development Efficiency](#)  [Foundational Services for SAP BTP](#).

To get notifications for updates and downtimes, subscribe at the [Cloud System Notification Subscriptions](#) application. Create a subscription by specifying Cloud Product, Cloud Service, and Notification Type. For more information, see [Cloud System Notification Subscriptions User Guide](#).

### ❖ Example

To receive notifications about changes in the IP ranges of a region, subscribe to ► [SAP Extension Suite – Development Efficiency](#) ► [Foundational Services for SAP BTP](#) ▶.

### ❖ Example

To receive notifications about what application processes to restart to apply new security patches, subscribe to ► [SAP Extension Suite – Development Efficiency](#) ► [Foundational Services for SAP BTP](#) ▶.

## Announcements and Subscriptions [China (Shanghai) Region]

You can follow the availability of the platform and the announcements about upcoming updates and downtimes at <https://status.cn40.platform.sapcloud.cn/> ↗. Subscribe to the status page to get notifications for updates and downtimes.

## Announcements and Subscriptions [Government Cloud (US) Region]

Planned downtimes and outage communication are sent through e-mail to the initial administrator of your global account.

## Related Information

[What's New](#)

[Regions \[page 17\]](#)

## 8.4 Operating Model

An operating model clearly defines the separation of tasks between SAP and the customer during all phases of an integration project.

The environment and its services have been developed on the assumption that specific processes and tasks are the responsibility of the customer. The operating model contains all processes and tasks involved in operating the platform and the services and specifies how the responsibilities are divided between SAP and the customer for each individual task. It doesn't include the operation of systems and devices residing at operational facilities owned by the customer or any other third party, as these are the customer's responsibility.

Changes to the operating model defined for the services in scope are published using the *What's New* section of the platform. Customers and other interested parties must review the product documentation on a regular

basis. If critical changes are made to the operating model, which require action on the customer side, an explicit notification is sent by e-mail to the affected customers.

It isn't the intent of this document to supplement or modify the contractual agreement between SAP and the customer for the purchase of any of the services in scope. In the event of a conflict, the contractual agreement between SAP and the customer as set out in the Order Form, the General Terms and Conditions, the supplemental terms and conditions, and any resources referenced by those documents always takes precedence over this document.

#### [Operating Model in the Cloud Foundry Environment \[page 3154\]](#)

This operating model clearly defines the separation of tasks between SAP and the customer during all phases of a project.

#### [Operating Model in the Kyma Environment \[page 3160\]](#)

This operating model clearly defines the separation of tasks between SAP and the customer during all phases of a project.

#### [Comparison between the Operating Models of Kyma and Cloud Foundry Runtimes \[page 3167\]](#)

This operating model clearly defines the separation of tasks between SAP and the customer. It's relevant for all phases of a project for both environments.

## 8.4.1 Operating Model in the Cloud Foundry Environment

This operating model clearly defines the separation of tasks between SAP and the customer during all phases of a project.

The responsibilities for operating the Cloud Foundry environment are listed in the following service catalog.

### Service Catalog

Process	Task	Responsibility
Communication Management	Appoint an English-speaking contact person and communicate the name to SAP. This is required to ensure timely processing of configuration change requests affecting the customer system, interacting with SAP for efficient case processing, and other interaction between SAP and the customer.	Customer
Communication Management	Subscribe to the communication channels offered by SAP for receiving prompt information about any service disruptions, critical maintenance activities affecting the customer system, and change requests requiring action on the customer side.	Customer

Process	Task	Responsibility
Communication Management	Inform the customer about service disruptions and critical maintenance activities affecting the customer system.	SAP
Asset Management	Manage the hardware and infrastructure resources in the region, from acquisition through disposal. This includes the request and approval process, procurement management, lifecycle management, and disposal management.	SAP
Asset Management	Protect IT assets such as systems, network, and data from threats that arise from unauthorized physical access or physical influence on those assets.	SAP
Provisioning	Provision resources and systems to customers in accordance with the ordered package and subscriptions. This includes the allocation and provisioning of technical (physical and virtual) resources, such as storage, network, compute units, systems, and database hosts, the deployment of SAP's application software and the proper initial configuration of quotas, service subscriptions, permissions, and trust configuration.	SAP
Provisioning	Provide a quota according to the ordered package and subscriptions that can be used to enable resources and services (for example, subscribing to a service).	Customer

Process	Task	Responsibility
Change Management	Apply regular product increments, as well as corrections to the infrastructure, systems, and services to avoid cases with minimal possible disruption of normal operations. Ensure that all platform changes (such as updates of the runtime or operating system patches, but not of the customer applications) are evaluated, authorized, prioritized, planned, tested, implemented, documented, and reviewed prior to implementation.	SAP
Change Management	Perform updates of the infrastructure, systems, and services in a biweekly cycle if required. Respectively, for selected services, offer self-services for applying controlled updates of new versions. Emergency changes, for example, triggered by Case Management processes, have accelerated testing, approval, and implementation.	SAP
Change Management	<ul style="list-style-type: none"> <li>• Ensure prompt delivery of security patches via the Security Patch Management process.</li> <li>• Provision new database systems and Java applications with the latest patched versions.</li> <li>• Apply the security patches on live customer systems (application runtimes or databases), in case the patches don't require downtime, or if the vulnerable system puts at risk SAP or other customers.</li> <li>• Inform the customers about the availability of security patches.</li> </ul>	SAP
Change Management	Adopt the latest patches or updates via the available self-services and by restarting applications when necessary. For example, when a security issue arises.	Customer

Process	Task	Responsibility
Case Management	Process cases reported by the customer according to the Service Level Agreement. The case is recorded and prioritized in the case tracking system. Monitor the status and progress of the case throughout its whole lifecycle and give regular status updates to the customer.	SAP
Case Management	In the event of cases, make reasonable effort to support end users and manage their cases, to explore self-help tools to find already documented solutions, and to liaise with SAP support in the event of new problems to ensure timely processing of cases affecting the resources in the customer account.	Customer
Case Management	Confirm case resolution in the case tracking system.	Customer
Service Requests	Process service requests reported by the customer according to the Service Level Agreement. The service request is recorded and prioritized in the service request tracking system. Monitor the status and progress of the service request throughout its whole lifecycle and give regular status updates to the customer.	SAP
Service Requests	Confirm service request completion in the service request tracking system.	Customer
Backup & Restore	Perform a backup of the database systems hosted in the subaccount. A database log backup is done according to the Service Level Agreement and stored on the primary storage. The logs are transferred from primary to secondary storage according to the Service Level Agreement. Full data backup is done every day.	SAP

Process	Task	Responsibility
Backup & Restore	Restore previously backed-up data to recover to a consistent state. Note: For some database services, there is a self-service for the restore process. Verify the completeness of the restored data based on log files created during the recovery and smoke tests to verify the system's consistency.	SAP
Backup & Restore	Give regular status updates to the customer throughout the entire restore procedure.	SAP
Backup & Restore	Collaborate with SAP to ensure timely processing of data restores if required.	Customer
Backup & Restore	Validate logical integrity and consistency of the restored data.	Customer
User Access Management	Manage users, permissions, and security configurations within the subaccount.	Customer
System Monitoring	Ensure availability of the customer system according to the Service Level Agreements as agreed in the contractual agreement between SAP and the customer, by active monitoring, prompt issue detection, and case prevention.	SAP
System Monitoring	Monitor the resource consumption (memory, CPU, storage) to detect issues in technical operations.	SAP
Malware Management	Ensure that the infrastructure and platform services are free of viruses, spam, spyware, and other malicious software. If malware is detected, an auto-notification is generated, which is assessed and resolved by SAP.	SAP

Process	Task	Responsibility
Application Management	Design, develop, deploy, configure, maintain, and operate the application within the subaccount. This includes maintaining a staged environment for application delivery (if required), application resource management, and managing application availability and performance.	Customer
Application Management	Provide infrastructure, tools, and application programming interfaces for the lifecycle management and operations of the application in the subaccount.	SAP
Application Management	Regularly adopt the latest versions of the tools for lifecycle management and operations offered at the <a href="#">SAP Development Tools site</a> .	Customer
Network Management	Manage the network isolation of the subaccounts provisioned to the customer.	SAP
Network Management	Operate the network infrastructure transparently for customers, ensuring elasticity, high availability, and security.	SAP
Network Management	Create and manage own Web domain for the application in the subaccount to ensure data isolation.	Customer
Penetration Testing	Inform SAP about any penetration testing that shall be performed for the customer account and ask for their approval. Testing isn't allowed on any resources shared with other customers. The results, if any, from the test are to be treated strictly as the confidential information of SAP and the customer aren't to be shared with any person or entity without explicit written authorization from SAP. Customers are required to share the results with SAP and work together with SAP's operations to mitigate or remedy any security issues.	Customer

Process	Task	Responsibility
Decommissioning	Ensure the secure deletion of data and hardware disposal. This includes the disassembling of systems along with peripherals and their removal from the region. Before dismantling and hand-over for further use or return to the vendor, the data is wiped securely from the system.	SAP

**Parent topic:** Operating Model [page 3153]

## Related Information

[Operating Model in the Kyma Environment \[page 3160\]](#)

[Comparison between the Operating Models of Kyma and Cloud Foundry Runtimes \[page 3167\]](#)

[SLAs for Cloud Services](#)

## 8.4.2 Operating Model in the Kyma Environment

This operating model clearly defines the separation of tasks between SAP and the customer during all phases of a project.

The responsibilities for operating the SAP BTP, Kyma runtime are listed in the following service catalog.

### ⓘ Note

The following information is valid for the managed offering, SAP BTP, Kyma runtime, not for the open source project "Kyma".

### Service Catalog

Process	Task	Responsibility
Communication Management	Appoint an English-speaking contact person and communicate the name to SAP. This is required to ensure timely processing of configuration change requests affecting the customer system, interacting with SAP for efficient case processing, and other interaction between SAP and the customer.	Customer

Process	Task	Responsibility
Communication Management	Subscribe to the communication channels offered by SAP for receiving prompt information about any service disruptions, critical maintenance activities affecting the customer system, and change requests requiring action on the customer side.	Customer
Communication Management	Inform the customer about service disruptions and critical maintenance activities affecting the customer system.	SAP
Asset Management	Manage the hardware and infrastructure resources in the region, from acquisition through disposal. This includes the request and approval process, procurement management, lifecycle management, and disposal management.	SAP
Asset Management	Protect IT assets such as systems, network, and data from threats that arise from unauthorized physical access or physical influence on those assets.	SAP
Provisioning	Provision resources and systems to customers in accordance with the ordered package and subscriptions. This includes the allocation and provisioning of technical (physical and virtual) resources, such as storage, network, compute units, systems, and database hosts, the deployment of SAP's application software and the proper initial configuration of quotas, service subscriptions, permissions, and trust configuration.	SAP

Process	Task	Responsibility
Provisioning	<p>Provide a quota according to the ordered package and subscriptions that can be used to enable resources and services (for example, subscribing to a service).</p> <p>Kyma runtime offers a consumption-based service plan. Hence, you can limit the resources (and costs) by configuring the minimum and maximum VMs. Such a configuration impacts the resources (compute units) available for developers to deploy applications.</p>	Customer
Change Management	<p>Apply regular product increments, as well as corrections to the infrastructure, systems, and services to avoid cases with minimal possible disruption of normal operations. Ensure that all platform changes (such as updates of the runtime or operating system patches, but not of the customer applications) are evaluated, authorized, prioritized, planned, tested, implemented, documented, and reviewed prior to implementation.</p>	SAP
Change Management	<p>Perform updates of the infrastructure, systems, and services if required. Respectively, for selected services, offer self-services for applying controlled updates of new versions. Emergency changes, for example, triggered by Case Management processes, have accelerated testing, approval, and implementation.</p>	SAP

Process	Task	Responsibility
Change Management	<ul style="list-style-type: none"> <li>• Ensure prompt delivery of security patches via the Security Patch Management process.</li> <li>• Apply the security patches on live customer systems (application runtimes or databases), in case the patches don't require downtime, or if the vulnerable system puts at risk SAP or other customers.</li> <li>• Inform the customers about the availability of security patches.</li> </ul>	SAP
Change Management	<ul style="list-style-type: none"> <li>• Handle any security patches to Java or any other languages. You have to create a new Docker image and update the applications to use the new image.</li> <li>• Adopt the latest patches or updates by restarting applications when necessary. For example, when a security issue arises.</li> </ul>	Customer
Case Management	Process cases reported by the customer according to the Service Level Agreement. The case is recorded and prioritized in the case tracking system. Monitor the status and progress of the case throughout its whole lifecycle and give regular status updates to the customer.	SAP
Case Management	In the event of cases, make reasonable effort to support end users and manage their cases, to explore self-help tools to find already documented solutions, and to liaise with SAP support in the event of new problems to ensure timely processing of cases affecting the resources in the customer account.	Customer
Case Management	Confirm case resolution in the case tracking system.	Customer

Process	Task	Responsibility
Service Requests	<p>Process service requests reported by the customer according to the Service Level Agreement. The service request is recorded and prioritized in the service request tracking system. Monitor the status and progress of the service request throughout its whole lifecycle and give regular status updates to the customer.</p>	SAP
Service Requests	Confirm service request completion in the service request tracking system.	Customer
Backup & Restore	<p>Perform a backup of the database systems hosted in the subaccount. A database log backup is done according to the Service Level Agreement and stored on the primary storage. The logs are transferred from primary to secondary storage according to the Service Level Agreement. Full data backup is done every day.</p> <p>Note: If you as a customer deploy your custom databases, you're then responsible. See <a href="#">Volume Backup for Customer Data</a>.</p>	SAP Customer
Backup & Restore	<p>Restore previously backed-up data to recover to a consistent state. Note: For some database services, there is a self-service for the restore process. Verify the completeness of the restored data based on log files created during the recovery and smoke tests to verify the system's consistency.</p> <p>Note: If you as a customer deploy your custom databases, you're then responsible. See <a href="#">Volume Backup for Customer Data</a>.</p>	SAP Customer

Process	Task	Responsibility
Backup & Restore	<p>Give regular status updates to the customer throughout the entire restore procedure.</p> <p>This responsibility is applicable only for the SAP BTP database services.</p> <p>Note: If you as a customer deploy your custom databases, you're then responsible. See <a href="#">Volume Backup for Customer Data</a>.</p>	SAP Customer
Backup & Restore	<p>Collaborate with SAP to ensure timely processing of data restores if required.</p> <p>This responsibility is applicable only for the SAP BTP database services.</p>	Customer
Backup & Restore	Validate logical integrity and consistency of the restored data.	Customer
User Access Management	Manage users, permissions, and security configurations within the subaccount.	Customer
System Monitoring	<p>Ensure availability of the customer system according to the Service Level Agreements as agreed in the contractual agreement between SAP and the customer, by active monitoring, prompt issue detection, and case prevention.</p>	SAP
System Monitoring	Monitor the resource consumption (memory, CPU, storage) to detect issues in technical operations.	SAP
Malware Management	<p>Ensure that the infrastructure and platform services are free of viruses, spam, spyware, and other malicious software. If malware is detected, an auto-notification is generated, which is assessed and resolved by SAP.</p> <p>Note: If your custom application has vulnerabilities, you as a customer are responsible to fix them, to build a new Docker image, and to update the application.</p>	SAP Customer

Process	Task	Responsibility
Application Management	Design, develop, deploy, configure, maintain, and operate the application within the subaccount. This includes maintaining a staged environment for application delivery (if required), application resource management, and managing application availability and performance.	Customer
Application Management	Provide infrastructure, tools, and application programming interfaces for the lifecycle management and operations of the application in the subaccount.	SAP
Application Management	Regularly adopt the latest versions of the tools for lifecycle management and operations offered at the <a href="#">SAP Development Tools site</a> .	Customer
Network Management	Manage the network isolation of the subaccounts provisioned to the customer.	SAP
Network Management	Operate the network infrastructure transparently for customers, ensuring elasticity, high availability, and security.	SAP
Network Management	Create and manage a custom Web domain for the application in the subaccount to ensure data isolation.	Customer
Network Management	Create and manage the default SAP Web domain for the application in the subaccount to ensure data isolation.	SAP

Process	Task	Responsibility
Penetration Testing	Inform SAP about any penetration testing that shall be performed for the customer account and ask for their approval. Testing isn't allowed on any resources shared with other customers. The results, if any, from the test are to be treated strictly as the confidential information of SAP and the customer aren't to be shared with any person or entity without explicit written authorization from SAP. Customers are required to share the results with SAP and work together with SAP's operations to mitigate or remedy any security issues.	Customer
Decommissioning	Ensure the secure deletion of data and hardware disposal. This includes the disassembling of systems along with peripherals and their removal from the region. Before dismantling and hand-over for further use or return to the vendor, the data is wiped securely from the system.	SAP

**Parent topic:** Operating Model [page 3153]

## Related Information

[Operating Model in the Cloud Foundry Environment \[page 3154\]](#)

[Comparison between the Operating Models of Kyma and Cloud Foundry Runtimes \[page 3167\]](#)

[SLAs for Cloud Services](#)

### 8.4.3 Comparison between the Operating Models of Kyma and Cloud Foundry Runtimes

This operating model clearly defines the separation of tasks between SAP and the customer. It's relevant for all phases of a project for both environments.

The responsibilities for operating the Cloud Foundry and Kyma runtimes are described in the following service catalog.

## Service Catalog

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Communication Management	Appoint an English-speaking contact person and communicate the name to SAP. This is required to ensure timely processing of configuration change requests affecting the customer system, interacting with SAP for efficient case processing, and other interaction between SAP and the customer.	Customer	Customer	No
Communication Management	Subscribe to the communication channels offered by SAP for receiving prompt information about any service disruptions, critical maintenance activities affecting the customer system, and change requests requiring action on the customer side.	Customer	Customer	No
Communication Management	Inform the customer about service disruptions and critical maintenance activities affecting the customer system.	SAP	SAP	No
Asset Management	Manage the hardware and infrastructure resources in the region, from acquisition through disposal. This includes the request and approval process, procurement management, lifecycle management, and disposal management.	SAP	SAP	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Asset Management	Protect IT assets such as systems, network, and data from threats that arise from unauthorized physical access or physical influence on those assets.	SAP	SAP	No
Provisioning	Provision resources and systems to customers in accordance with the ordered package and subscriptions. This includes the allocation and provisioning of technical (physical and virtual) resources, such as storage, network, compute units, systems, and database hosts, the deployment of SAP's application software and the proper initial configuration of quotas, service subscriptions, permissions, and trust configuration.	SAP	SAP	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Provisioning	<p>Provide a quota according to the ordered package and subscriptions that can be used to enable resources and services (for example, subscribing to a service).</p> <p>Kyma runtime doesn't offer a subscription-based service plan, but a consumption-based one. However, you can limit the resources (and costs) in other ways.</p> <p>What you can still configure is the minimum and maximum VMs for a Kyma runtime. This impacts the resources (compute units) available in a Kyma runtime for developers to deploy applications.</p>	Customer	Customer	Yes
Change Management	Apply regular product increments, as well as corrections to the infrastructure, systems, and services to avoid cases with minimal possible disruption of normal operations. Ensure that all platform changes (such as updates of the runtime or operating system patches, but not of the customer applications) are evaluated, authorized, prioritized, planned, tested, implemented, documented, and reviewed prior to implementation.	SAP	SAP	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Change Management	<p>Perform updates of the infrastructure, systems, and services if required. Respectively, for selected services, offer self-services for applying controlled updates of new versions. Emergency changes, for example, triggered by Case Management processes, have accelerated testing, approval, and implementation.</p> <p>The updates for the Cloud Foundry runtime are performed in bi-weekly cycles.</p>	SAP	SAP	Yes

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Change Management	<ul style="list-style-type: none"> <li>Ensure prompt delivery of security patches via the Security Patch Management process.</li> <li>Provision new database systems and Java applications with the latest patched versions in the Cloud Foundry runtime.</li> </ul> <p>In the Kyma runtime, you as a customer are responsible for handling any security patches to Java or any other languages. You have to create a new Docker image and update the applications running in the Kyma runtime to use the new image.</p> <ul style="list-style-type: none"> <li>Apply the security patches on live customer systems (application runtimes or databases), in case the patches don't require downtime, or if the vulnerable system puts at risk SAP or other customers.</li> <li>Inform the customers about the availability of security patches.</li> </ul>	SAP	SAP Customer	Yes

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Change Management	<p>Adopt the latest patches or updates by restarting applications when necessary. For example, when a security issue arises.</p> <p>In the Cloud Foundry runtime, you can also adopt the latest patches or updates via the available self-services.</p>	Customer	Customer	Yes
Case Management	<p>Process cases reported by the customer according to the Service Level Agreement. The case is recorded and prioritized in the case tracking system. Monitor the status and progress of the case throughout its whole lifecycle and give regular status updates to the customer.</p>	SAP	SAP	No
Case Management	<p>In the event of cases, make reasonable effort to support end users and manage their cases, to explore self-help tools to find already documented solutions, and to liaise with SAP support in the event of new problems to ensure timely processing of cases affecting the resources in the customer account.</p>	Customer	Customer	No
Case Management	<p>Confirm case resolution in the case tracking system.</p>	Customer	Customer	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Service Requests	Process service requests reported by the customer according to the Service Level Agreement. The service request is recorded and prioritized in the service request tracking system. Monitor the status and progress of the service request throughout its whole lifecycle and give regular status updates to the customer.	SAP	SAP	No
Service Requests	Confirm service request completion in the service request tracking system.	Customer	Customer	No
Backup & Restore	Perform a backup of the database systems hosted in the subaccount. A database log backup is done according to the Service Level Agreement and stored on the primary storage. The logs are transferred from primary to secondary storage according to the Service Level Agreement. Full data backup is done every day.  Note: If you as a customer deploy your custom databases in the Kyma runtime, you're then responsible. See <a href="#">Volume Backup for Customer Data</a> .	SAP	SAP Customer	Yes

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Backup & Restore	<p>Restore previously backed-up data to recover to a consistent state. Note: For some database services, there's a self-service for the restore process. Verify the completeness of the restored data based on log files created during the recovery and smoke tests to verify the system's consistency.</p> <p>Note: If you as a customer deploy your custom databases in the Kyma runtime, you're then responsible. See <a href="#">Volume Backup for Customer Data</a>.</p>	SAP	SAP Customer	Yes
Backup & Restore	<p>Give regular status updates to the customer throughout the entire restore procedure.</p> <p>In the Kyma runtime, this responsibility is applicable only for the SAP BTP database services.</p> <p>Note: If you as a customer deploy your custom databases in the Kyma runtime, you're then responsible. See <a href="#">Volume Backup for Customer Data</a>.</p>	SAP	SAP Customer	Yes

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Backup & Restore	<p>Collaborate with SAP to ensure timely processing of data restores if required.</p> <p>In the Kyma runtime, this responsibility is applicable only for the SAP BTP database services.</p>	Customer	Customer	Yes
Backup & Restore	Validate logical integrity and consistency of the restored data.	Customer	Customer	No
User Access Management	Manage users, permissions, and security configurations within the subaccount.	Customer	Customer	No
System Monitoring	Ensure availability of the customer system according to the Service Level Agreements as agreed in the contractual agreement between SAP and the customer, by active monitoring, prompt issue detection, and case prevention.	SAP	SAP	No
System Monitoring	Monitor the resource consumption (memory, CPU, storage) to detect issues in technical operations.	SAP	SAP	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Malware Management	<p>Ensure that the infrastructure and platform services are free of viruses, spam, spyware, and other malicious software. If malware is detected, an auto-notification is generated, which is assessed and resolved by SAP.</p> <p>If your custom application in the Kyma runtime has vulnerabilities, you as a customer are responsible to fix them, to build a new Docker image, and to update the application.</p>	SAP	SAP Customer	Yes
Application Management	Design, develop, deploy, configure, maintain, and operate the application within the subaccount. This includes maintaining a staged environment for application delivery (if required), application resource management, and managing application availability and performance.	Customer	Customer	No
Application Management	Provide infrastructure, tools, and application programming interfaces for the lifecycle management and operations of the application in the subaccount.	SAP	SAP	No

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Application Management	Regularly adopt the latest versions of the tools for lifecycle management and operations offered at the <a href="#">SAP Development Tools site</a> .	Customer	Customer	No
Network Management	Manage the network isolation of the subaccounts provisioned to the customer.	SAP	SAP	No
Network Management	Operate the network infrastructure transparently for customers, ensuring elasticity, high availability, and security.	SAP	SAP	No
Network Management	Create and manage a custom Web domain for the application in the subaccount to ensure data isolation.  In the Kyma runtime, if you as a customer bring your own domain to expose workloads, you're then responsible. However, if you use the default SAP domain, SAP's responsible.	Customer	SAP Customer	Yes

Process	Task	Responsibility in Cloud Foundry	Responsibility in Kyma	Differences
Penetration Testing	<p>Inform SAP about any penetration testing that shall be performed for the customer account and ask for their approval. Testing isn't allowed on any resources shared with other customers.</p> <p>The results, if any, from the test are to be treated strictly as the confidential information of SAP and the customer aren't to be shared with any person or entity without explicit written authorization from SAP. Customers are required to share the results with SAP and work together with SAP's operations to mitigate or remedy any security issues.</p>	Customer	Customer	No
Decommissioning	<p>Ensure the secure deletion of data and hardware disposal. This includes the disassembling of systems along with peripherals and their removal from the region. Before dismantling and handover for further use or return to the vendor, the data is wiped securely from the system.</p>	SAP	SAP	No

**Parent topic:** Operating Model [page 3153]

## Related Information

[Operating Model in the Cloud Foundry Environment \[page 3154\]](#)

## 8.5 Support Components

A list of support components for SAP BTP services and tools. Filter for the service you want to find the component for or have a look at the tools and software logistics section.

### ⓘ Note

The table below lists the support components for services. If you are looking for components of tools or issues related to software logistics, see [Additional Components \[page 3263\]](#).

## Service Components

Service Availability

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Feature Flags Service	Control the rollout of new features.	BC-CP-CF-FEATUREFLG	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Australia (Sydney) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Feature Flags Service	Control the rollout of new features.	BC-CP-CF-FEATUREFLG	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore	Yes	Available
Feature Flags Service	Control the rollout of new features.	BC-CP-CF-FEATUREFLG	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt)	Yes	Available
Feature Flags Service	Control the rollout of new features.	BC-CP-CF-FEATUREFLG	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Dynatrace Agent Activation	Connect your Java applications to a Dynatrace SaaS monitoring environment.	BC-NEO-MON-APM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Conversational AI	Build and deploy innovative chatbots using this comprehensive end-to-end platform.	CA-ML-CAI	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
Application Logging Service	Create, store, access, and analyze application logs.	BC-NEO-LOG	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Application Logging Service	Create, store, access, and analyze application logs.	BC-CP-CF-APPLOG	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Application Logging Service	Create, store, access, and analyze application logs.	BC-CP-CF-APPLOG	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
Application Logging Service	Create, store, access, and analyze application logs.	BC-CP-CF-APPLOG	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Application Logging Service	Create, store, access, and analyze application logs.	BC-CP-CF-APPLOG	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Open Connectors	Simplify integration via APIs.	LOD-OCN-OPS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US West (Chandler) US West (Colorado Springs)	Yes	Available
Open Connectors	Simplify integration via APIs.	LOD-OCN-OPS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Canada (Montreal) Brazil (São Paulo) Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Open Connectors	Simplify integration via APIs.	LOD-OCN-OPS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
Open Connectors	Simplify integration via APIs.	LOD-OCN-OPS	Cloud Foundry	GCP	US Central (IA)	Yes	Available
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore Japan (Tokyo) Australia (Sydney) Brazil (São Paulo) Canada (Montreal)	Yes	Available
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POSTGRES	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POSTGRES	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POSTGRES	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RABBITMQ	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RABBITMQ	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RABBITMQ	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
OData Provisioning	Access data in SAP Business Suite using OData services.	OPU-GW-OD-FW	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) KSA (Riyadh) Canada (Toronto) Brazil (São Paulo)	Yes	Available
OData Provisioning	Access data in SAP Business Suite using OData services.	OPU-GW-OD-FW	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
OData Provisioning	Access data in SAP Business Suite using OData services.	OPU-GW-OD-FW	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Git Service	Store and version source code in Git repositories.	BC-NEO-GIT	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Java Server	Develop and run Java Web applications on SAP Cloud Platform Neo Environment.	BC-NEO-RT-JAV	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
SAP Analytics Cloud, embedded edition	Analyze data via live connection to your business application's SAP HANA database.	LOD-ANA-OEM-CP	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore	Yes	Available
Data Attribute Recommendation	Apply machine learning to predict and classify data records.	CA-ML-DAR	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Australia (Sydney) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Document Service	Store and manage your documents.	BC-NEO-ECM-DS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
SAP HANA Cloud	A single gateway to all your data.	HAN-CLS-HC	Cloud Foundry	AWS	Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP HANA Cloud	A single gateway to all your data.	HAN-CLS-HC	Cloud Foundry	Azure	Australia (Sydney) Singapore Switzerland (Zurich) EU Access Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available
SAP HANA Cloud	A single gateway to all your data.	HAN-CLS-HC	Cloud Foundry	GCP	Europe (Frankfurt) India (Mumbai) US Central (IA)	Yes	Available
Business Entity Recognition	Detect and highlight entities from unstructured text using machine learning.	CA-ML-BER	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Australia (Sydney) Europe (Frankfurt) EU Access	Yes	Available
SAP S/4HANA Cloud Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP S/4HANA Cloud systems.	BC-NEO-EXT-S4C	Cloud Foundry	AWS	Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP S/4HANA Cloud Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP S/4HANA Cloud systems.	BC-NEO-EXT-S4C	Cloud Foundry	Azure	Australia (Sydney) Singapore Brazil (São Paulo) Switzerland (Zurich) EU Access Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available
SAP S/4HANA Cloud Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP S/4HANA Cloud systems.	BC-NEO-EXT-S4C	Cloud Foundry	GCP	Australia (Sydney) Europe (Frankfurt) India (Mumbai) US Central (IA)	Yes	Available
Identity Authentication - Additional Tenant	Secure authentication and single sign-on for users in the cloud.	BC-IAM-IDS		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Solution Life-cycle Management Service	Deploy, subscribe and transport Solutions using Multi-Target Applications (MTA).	BC-NEO-LCM-SRV	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) Australia (Sydney) Japan (Tokyo) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
SAP Procurement Intelligence	Automate and simplify your business processes by applying machine learning intelligence.	CA-ML-PA	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
Document Classification	Classify business documents automatically using machine learning.	CA-ML-BDP	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Cloud Transport Management	Manage transports of development artifacts and application-specific content.	BC-CP-LCM-TMS	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal) Brazil (São Paulo) Japan (Tokyo)	Yes	Available
Cloud Transport Management	Manage transports of development artifacts and application-specific content.	BC-CP-LCM-TMS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore Japan (Tokyo) Switzerland (Zurich) EU Access	Yes	Available
Cloud Transport Management	Manage transports of development artifacts and application-specific content.	BC-CP-LCM-TMS	Cloud Foundry	GCP	Europe (Frankfurt) US Central (IA) India (Mumbai) KSA (Dammam) public sector Israel (Tel Aviv)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Data Enrichment Service	Create or enrich master data using trusted third-party data.	LOD-MDM-DE	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-BAS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Canada (Montreal) Japan (Tokyo) Brazil (São Paulo) Singapore Europe (Frankfurt) EU Access South Korea (Seoul)	Yes	Available
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-BAS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-BAS	Cloud Foundry	GCP	US Central (IA) Israel (Tel Aviv) India (Mumbai) KSA (Dammam) public sector Australia (Sydney) Europe (Frankfurt)	Yes	Available
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-BAS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Data Enrichment Service	Create or enrich master data using trusted third-party data.	LOD-MDM-DE	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available
Service Ticket Intelligence	Build a self-driven customer service powered by machine learning.	CA-ML-STI	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access	Yes	Available
Service Ticket Intelligence	Build a self-driven customer service powered by machine learning.	CA-ML-STI	Cloud Foundry	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-REDIS	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-REDIS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-REDIS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Identity Authentication	Secure authentication and single sign-on for users in the cloud.	BC-IAM-IDS		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Print Service	Manage print queues, connect print clients, and monitor print status.	BC-CCM-PRN-OM-SCP	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
SAP Intelligent Product Design	Generate, analyze and validate product requirements using intelligent tools.	PLM-DC	Neo	SAP	Europe (Rot)* Europe (Frankfurt)		Available
SAP Intelligent Product Design	Generate, analyze and validate product requirements using intelligent tools.	PLM-DC	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
SAP Real-Spend	Manage and track expenses, and compare them against your budget in real time.	LOD-MAP-RS	Neo	SAP	Europe (Rot)* Europe (Frankfurt)		Available
Variant Configuration	Configure your SAP ERP or SAP S/4HANA products interactively in the cloud	LOD-CPS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available
Variant Configuration	Configure your SAP ERP or SAP S/4HANA products interactively in the cloud	LOD-CPS	Cloud Foundry	Azure	Europe (Netherlands)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Document Management Service, Integration Option	Build API and UI based document management capabilities for your business apps.	BC-CP-CF-SDM	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) EU Access South Korea (Seoul)	Yes	Available
Document Management Service, Integration Option	Build API and UI based document management capabilities for your business apps.	BC-CP-CF-SDM	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore Australia (Sydney)	Yes	Available
Document Management Service, Integration Option	Build API and UI based document management capabilities for your business apps.	BC-CP-CF-SDM	Cloud Foundry	GCP	Europe (Frankfurt) US Central (IA)	Yes	Available
Document Management Service, Integration Option	Build API and UI based document management capabilities for your business apps.	BC-CP-CF-SDM	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP IoT Connect 365	Simplify the complex connectivity, scalability, and management of IoT.	BC-NEO-SVC-IOT		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available
Invoicing option for Pep- pol	Meet compliance requirements by exchanging documents with the Pep-pol network.	LOD-LH-DCS-PAP	Neo	SAP	Europe (Rot)*		Available
Serverless Runtime	Use extensions on SAP Business Technology Platform and get runtime access to OData services in SAP Integration Suite.	BC-CP-XF-SRT	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Japan (To-kyo) Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal)	Yes	Available
Serverless Runtime	Use extensions on SAP Business Technology Platform and get runtime access to OData services in SAP Integration Suite.	BC-CP-XF-SRT	Cloud Foundry	Azure	Europe (Nether-lands) US West (WA) US East (VA) Japan (To-kyo) Singapore	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP SuccessFactors Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP SuccessFactors systems.	BC-NEO-EXT-SF	Cloud Foundry	AWS	Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA)	Yes	Available
SAP SuccessFactors Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP SuccessFactors systems.	BC-NEO-EXT-SF	Cloud Foundry	Azure	Australia (Sydney) Singapore Brazil (São Paulo) Switzerland (Zurich) EU Access Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available
SAP SuccessFactors Extensibility	Automates the connectivity configuration of extension applications running on SAP BTP to SAP SuccessFactors systems.	BC-NEO-EXT-SF	Cloud Foundry	GCP	Australia (Sydney) Europe (Frankfurt) India (Mumbai) US Central (IA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Web IDE Full-Stack	Create and extend SAP full-stack applications for browsers and mobile devices.	CA-WDE-PLFRM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) UAE (Dubai) KSA (Riyadh)	Yes	Available
Mobile Service for App and Device Management	Manage your mobile devices.	MOB-SEC	Neo	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) US East (Sterling)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Singapore Japan (Tokyo) Brazil (São Paulo) South Korea (Seoul) Europe (Frankfurt) EU Access Canada (Montreal)	Yes	Available
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	GCP	US Central (IA) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Java Debugging	Debug your Java application even through networks with high latency.	BC-JVM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Personalized Recommendation	Personalized Recommendation draws insights from user data to deliver highly personalized recommendations.	CA-ML-PR	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP HANA Service	Create and consume SAP HANA databases.	BC-NEO-PERS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	GCP	US Central (IA)	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Object Store	Store and manage the blobs/objects on SAP BTP.	BC-CP-CF-OSAAS	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Object Store	Store and manage the blobs/objects on SAP BTP.	BC-CP-CF-OSAAS	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore	Yes	Available
Object Store	Store and manage the blobs/objects on SAP BTP.	BC-CP-CF-OSAAS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
SAP HANA spatial services	SAP HANA spatial services provides a set of APIs for location-based services.	BC-CP-CF-HSS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	AWS	Europe (Frankfurt) Brazil (São Paulo) Canada (Montreal) US East (VA) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo) Singapore US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt)	Yes	Available
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Job Scheduling Service	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-CP-CF-JBS	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available
Job Scheduling Service	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-CP-CF-JBS	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Job Scheduling Service	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-CP-CF-JBS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Job Scheduling Service	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-CP-CF-JBS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Credential Store	Store and retrieve credentials such as crypto-graphic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Brazil (São Paulo) Singapore Canada (Montreal) Japan (Tokyo)	Yes	Available
Credential Store	Store and retrieve credentials such as crypto-graphic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Credential Store	Store and retrieve credentials such as crypto-graphic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	GCP	Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Credential Store	Store and retrieve credentials such as cryptographic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Pricing service	Calculate prices for configurable- and non-configurable products	LOD-CPS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available
Pricing service	Calculate prices for configurable- and non-configurable products	LOD-CPS	Cloud Foundry	Azure	Europe (Netherlands)	Yes	Available
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Canada (Montreal) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	GCP	Israel (Tel Aviv)	Yes	Available
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
PostgreSQL, Hyperscaler Option	Create and consume relational databases with PostgreSQL based on hyperscalers.	BC-CP-BSB-POST-GRESQL	Cloud Foundry	AWS	Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA)	Yes	Available
PostgreSQL, Hyperscaler Option	Create and consume relational databases with PostgreSQL based on hyperscalers.	BC-CP-BSB-POST-GRESQL	Cloud Foundry	Azure	Australia (Sydney) Singapore Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
PostgreSQL, Hyperscaler Option	Create and consume relational databases with PostgreSQL based on hyperscalers.	BC-CP-BSB-POST-GRESQL	Cloud Foundry	GCP	Israel (Tel Aviv)	Yes	Available
Document Information Extraction	Use machine learning to automate your document information extraction processes.	CA-ML-BDP	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA) Australia (Sydney)	Yes	Available
SAP HANA spatial services	SAP HANA spatial services provides a set of APIs for location-based services.	BC-CP-CF-HSS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Cloud Integration for Data Services	Integrate data between on-premise and cloud applications on a scheduled (batch) basis.	LOD-HCI-DS	Neo	SAP	UAE (Dubai) Australia (Sydney) Europe (Rot)* Japan (Tokyo) KSA (Riyadh) US West (Colorado Springs)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
HTML5 Application Repository Service	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal) Japan (Tokyo)	Yes	Available
HTML5 Application Repository Service	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
HTML5 Application Repository Service	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	GCP	US Central (IA) Israel (Tel Aviv)	Yes	Available
HTML5 Application Repository Service	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Java Profiling	Profile and analyze your Java applications.	BC-JVM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
OAuth 2.0	Protect applications and APIs with OAuth 2.0.	BC-NEO-SEC-IAM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Identity Provisioning	Manage identity lifecycle processes for cloud and on-premise systems.	BC-IAM-IPS	Neo	SAP	UAE (Dubai) Australia (Sydney) Brazil (São Paulo) Canada (Toronto) Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) Japan (Tokyo) KSA (Riyadh) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs)	Yes	Available
Mobile Service for SAP Fiori	Optimize, build, manage, and monitor SAP Fiori apps on mobile devices.	MOB-FM	Neo	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) US East (Sterling) Canada (Toronto) Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Document Center	Use uniform standard-based file access and mobilize your business content.	BC-NEO-ECM-APP	Neo	SAP	Europe (Rot)* Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Cloud Portal Service	Create role based, multi-channel sites to access business apps and content.	EP-CPP-NEO-LS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) Europe (Amsterdam) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Cloud Portal Service	Create role based, multi-channel sites to access business apps and content.	EP-WZ-SRB	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal) Europe (Frankfurt) EU Access South Korea (Seoul)	Yes	Available
Cloud Portal Service	Create role based, multi-channel sites to access business apps and content.	EP-WZ-SRB	Cloud Foundry	Azure	Singapore US West (WA) US East (VA) Europe (Netherlands) Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Cloud Portal Service	Create role based, multi-channel sites to access business apps and content.	EP-WZ-SRB	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Cloud Portal Service	Create role based, multi-channel sites to access business apps and content.	EP-WZ-SRB	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Japan (Tokyo) Brazil (São Paulo) South Korea (Seoul) Europe (Frankfurt) EU Access Canada (Montreal)	Yes	Available
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
SAP ASE Service	Create and consume SAP ASE databases.	BC-NEO-PERS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Redis, Hyperscaler Option	Create and consume in-memory cache with Redis based on hyperscalers.	BC-CP-BSB-REDIS	Cloud Foundry	AWS	US East (VA) Japan (Tokyo) Europe (Frankfurt) EU Access Europe (Frankfurt) Canada (Montreal) US East (VA) Brazil (São Paulo) Singapore Australia (Sydney)	Yes	Available
Redis, Hyperscaler Option	Create and consume in-memory cache with Redis based on hyperscalers.	BC-CP-BSB-REDIS	Cloud Foundry	Azure	US East (VA) US West (WA) Japan (Tokyo) Europe (Netherlands) Singapore Australia (Sydney)	Yes	Available
Redis, Hyperscaler Option	Create and consume in-memory cache with Redis based on hyperscalers.	BC-CP-BSB-REDIS	Cloud Foundry	GCP	Israel (Tel Aviv)	Yes	Available
Administration Service	Administer the configuration and pricing services	LOD-CPS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available
Administration Service	Administer the configuration and pricing services	LOD-CPS	Cloud Foundry	Azure	Europe (Netherlands)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-UI5-FL-CLS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Singapore South Korea (Seoul) Australia (Sydney) Canada (Montreal) Europe (Frankfurt) EU Access Brazil (São Paulo)	Yes	Available
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-UI5-FL-CLS	Cloud Foundry	Azure	Europe (Netherlands) Australia (Sydney) US West (WA) US East (VA) Singapore Japan (Tokyo) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-UI5-FL-CLS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai) KSA (Dammam) public sector Israel (Tel Aviv) Australia (Sydney)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Cloud Logging	Store and analyze application logs, metrics, and traces based on Open-Search.	BC-CP-CLS	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal) US East (VA)	Yes	Available
Cloud Logging	Store and analyze application logs, metrics, and traces based on Open-Search.	BC-CP-CLS	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore	Yes	Available
Cloud Logging	Store and analyze application logs, metrics, and traces based on Open-Search.	BC-CP-CLS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai)	Yes	Available
Cloud Logging	Store and analyze application logs, metrics, and traces based on Open-Search.	BC-CP-CLS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Landscape Management Cloud	Manage your SAP systems in infrastructure as a service (IaaS) environments.	BC-VCM-LMC	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
Event Mesh	Connect applications, services and systems across different landscapes.	BC-CP-EM-MES	Cloud Foundry	AWS	US East (VA) Australia (Sydney) Singapore Brazil (São Paulo) Japan (Tokyo) Europe (Frankfurt) Canada (Montreal)	Yes	Available
Event Mesh	Connect applications, services and systems across different landscapes.	BC-CP-EM-MES	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Event Mesh	Connect applications, services and systems across different landscapes.	BC-CP-EM-MES	Cloud Foundry	GCP	Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Event Mesh	Connect applications, services and systems across different landscapes.	BC-CP-EM-MES	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Process Integration	Seamlessly orchestrate business processes and integrate data in real time.	LOD-HCI-PI-OPS-PROV	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) KSA (Riyadh) UAE (Dubai)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
<a href="#">Virtual Machine Service</a>	Virtualized hardware resources (CPU, RAM, disk space, installed OS) to install and maintain the Linux-based software.	BC-NEO-INFR-VM	Neo	SAP	Europe (Rot)* Europe (Amsterdam) Europe (Frankfurt) Australia (Sydney) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh) Japan (Tokyo) Canada (Toronto)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Connectivity Service	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Connectivity Service	Establish connections between cloud applications and on-premise systems.	BC-CP-CON-CF	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Connectivity Service	Establish connections between cloud applications and on-premise systems.	BC-CP-CON-CF	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
Connectivity Service	Establish connections between cloud applications and on-premise systems.	BC-CP-CON-CF	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Connectivity Service	Establish connections between cloud applications and on-premise systems.	BC-CP-CON-CF	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Platform Identity Provider Service	Use your user base from the identity authentication tenant for admin tasks.	BC-NEO-SEC-IAM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Forms Service by Adobe	Generate print and interactive forms using Adobe Document Services.	BC-SRV-FP-NEO	Neo	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) UAE (Dubai) Europe (Amsterdam) KSA (Riyadh) Europe (Frankfurt) US West (Colorado Springs)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Fiori Cloud	Revamp your user experience with SAP Fiori on SAP Cloud Platform.	EP-CPP-NEO-OPS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) Europe (Amsterdam) UAE (Dubai) KSA (Riyadh)	Yes	Available
Data Quality Services	Embed data quality services to validate addresses and enrich with geocodes.	EIM-DQM-SVS	Neo	SAP	Europe (Rot)* Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Authorization and Trust Management Service	Manage application authorizations and trusted connections to identity providers.	BC-NEO-SEC-IAM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Authorization and Trust Management Service	Manage application authorizations and trusted connections to identity providers.	BC-CP-CF-SEC-IAM	Cloud Foundry	AWS	US East (VA) Europe (Frankfurt) Europe (Frankfurt) EU Access Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Authorization and Trust Management Service	Manage application authorizations and trusted connections to identity providers.	BC-CP-CF-SEC-IAM	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore Japan (Tokyo)	Yes	Available
Authorization and Trust Management Service	Manage application authorizations and trusted connections to identity providers.	BC-CP-CF-SEC-IAM	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Authorization and Trust Management Service	Manage application authorizations and trusted connections to identity providers.	BC-CP-CF-SEC-IAM	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
SAP Customer Order Sourcing	Calculate sourcing results based on your own sourcing strategies.	LOD-CID-OSR	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Java Apps Lifecycle Management	Manage the lifecycle of Java applications by using a REST API.	BC-NEO-DPL	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Gamification	Introduce gamification concepts into your applications.	BC-NEO-SVC-GAM	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AUTOSCALE	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AUTOSCALE	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AUTOSCALE	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AUTOSCALE	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
SAP Build Work Zone, standard edition	Simplify access to applications by establishing a central entry point.	EP-WZ-SRB	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal) Europe (Frankfurt) EU Access South Korea (Seoul)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Build Work Zone, standard edition	Simplify access to applications by establishing a central entry point.	EP-WZ-SRB	Cloud Foundry	Azure	Singapore US West (WA) US East (VA) Europe (Netherlands) Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
SAP Build Work Zone, standard edition	Simplify access to applications by establishing a central entry point.	EP-WZ-SRB	Cloud Foundry	GCP	US Central (IA) India (Mumbai) Israel (Tel Aviv) Europe (Frankfurt) KSA (Dammam) public sector Australia (Sydney)	Yes	Available
SAP Build Work Zone, standard edition	Simplify access to applications by establishing a central entry point.	EP-WZ-SRB	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Smart Business Service	Expose KPIs and OPIs as SAP Fiori applications without the need to write any code.	CA-GTF-SB-HCP	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai)	Yes	Available
SAP Smart Business Service	Expose KPIs and OPIs as SAP Fiori applications without the need to write any code.	CA-GTF-SB-HCP	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
SAP Intelligent Robotic Process Automation	Design, configure, and execute automation projects.	CA-ML-IPA	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Australia (Sydney) Japan (Tokyo) US East (VA)	Yes	Available
SAP Intelligent Robotic Process Automation	Design, configure, and execute automation projects.	CA-ML-IPA	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Internet of Things	Develop, customize, and operate IoT business applications in the cloud.	BC-NEO-SVC-IOT	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Alert Notification	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Japan (Tokyo) Canada (Toronto) Australia (Sydney) KSA (Riyadh) UAE (Dubai) Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Alert Notification	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Singapore Australia (Sydney) Canada (Montreal) Brazil (São Paulo) Europe (Frankfurt) EU Access South Korea (Seoul)	Yes	Available
Alert Notification	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access	Yes	Available
Alert Notification	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai) Israel (Tel Aviv)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Leonardo ML Foundation	Infuse your applications with intelligent, easy-to-use services based on Machine Learning.	CA-ML-PLT	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Japan (Tokyo)	Yes	Available
SAP Leonardo ML Foundation	Infuse your applications with intelligent, easy-to-use services based on Machine Learning.	CA-ML-PLT	Cloud Foundry	GCP	US Central (IA)	Yes	Available
Monitoring Service	Monitor Java, SAP HANA XS and HTML5 applications, and databases.	BC-NEO-MON	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Continuous Integration and Delivery	Configure and run pre-defined pipelines for continuous integration and delivery.	BC-CP-CF-CICD	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Japan (Tokyo) Australia (Sydney) South Korea (Seoul) Brazil (São Paulo) Singapore	Yes	Available
Continuous Integration and Delivery	Configure and run pre-defined pipelines for continuous integration and delivery.	BC-CP-CF-CICD	Cloud Foundry	Azure	Europe (Netherlands) US East (VA) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Continuous Integration and Delivery	Configure and run pre-defined pipelines for continuous integration and delivery.	BC-CP-CF-CICD	Cloud Foundry	GCP	Europe (Frankfurt) India (Mumbai) Israel (Tel Aviv) US Central (IA) KSA (Dammam) public sector Australia (Sydney)	Yes	Available
Web Analytics	Analyze usage of your websites and web applications.	CA-SWA	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Brazil (São Paulo) Japan (Tokyo) Singapore	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Web Analytics	Analyze usage of your websites and web applications.	CA-SWA	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
Data Retention Manager	Manage retention and residence rules to block or delete personal data.	LOD-GDP-RM	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney)	Yes	Available
Data Retention Manager	Manage retention and residence rules to block or delete personal data.	LOD-GDP-RM	Cloud Foundry	Azure	Europe (Netherlands) US West (WA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) KSA (Riyadh) UAE (Dubai)	Yes	Available
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	AWS	Australia (Sydney) Brazil (São Paulo) Japan (Tokyo) US East (VA) Europe (Frankfurt) Singapore Canada (Montreal) South Korea (Seoul) Europe (Frankfurt) EU Access	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo) US East (VA) Singapore Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	GCP	Europe (Frankfurt) US Central (IA) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Translation Hub	Translate UI texts and get suggestions for UI texts during development.	LOD-TH	Neo	SAP	Europe (Rot)* Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Europe (Frankfurt)	Yes	Available
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Neo	SAP	Europe (Rot)* US East (Sterling)		Available
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Cloud Foundry	Azure	US West (WA)	Yes	Available
SAP Data Intelligence	Connect data from various sources to enable next-generation data management.	CA-DI	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP Data Intelligence	Connect data from various sources to enable next-generation data management.	CA-DI	Cloud Foundry	Azure	Europe (Netherlands) US West (WA)	Yes	Available
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Canada (Toronto) Brazil (São Paulo) Australia (Sydney) Japan (Tokyo) UAE (Dubai) KSA (Riyadh)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Singapore Japan (Tokyo) Brazil (São Paulo) South Korea (Seoul) Europe (Frankfurt) EU Access Canada (Montreal)	Yes	Available
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney) Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Kyma Runtime	Develop and run containerized applications and extensions on Kubernetes.	BC-CP-XF-KYMA	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Europe (Frankfurt) EU Access Brazil (São Paulo) Australia (Sydney) Japan (Tokyo) Canada (Montreal) Singapore South Korea (Seoul)	Yes	Available
Kyma Runtime	Develop and run containerized applications and extensions on Kubernetes.	BC-CP-XF-KYMA	Cloud Foundry	GCP	Israel (Tel Aviv)	Yes	Available
Audit Log Service	A core, security and compliance based SAP BTP service to provide means for audit purposes.	BC-CP-CF-SEC-AU-DITLG	Cloud Foundry	AWS	Europe (Frankfurt) EU Access	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Audit Log Service	A core, security and compliance based SAP BTP service to provide means for audit purposes.	BC-CP-CF-SEC-AU-DITLG	Cloud Foundry	GCP	Israel (Tel Aviv)	Yes	Available
SAP IoT	Put raw sensor data into business context and leverage it in analytical or transactional applications.	IOT-BSV-APB	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
SAP IoT	Put raw sensor data into business context and leverage it in analytical or transactional applications.	IOT-BSV-APB	Cloud Foundry	Azure	Europe (Netherlands)	Yes	Available
Cloud Foundry Runtime	Operate polyglot cloud applications in Cloud Foundry.	BC-CP-CF	Cloud Foundry	AWS	US East (VA) Europe (Frankfurt) Europe (Frankfurt) EU Access Brazil (São Paulo) Japan (Tokyo) Canada (Montreal) Australia (Sydney) Singapore South Korea (Seoul)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Cloud Foundry Runtime	Operate polyglot cloud applications in Cloud Foundry.	BC-CP-CF	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo) Australia (Sydney)	Yes	Available
Cloud Foundry Runtime	Operate polyglot cloud applications in Cloud Foundry.	BC-CP-CF	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Cloud Foundry Runtime	Operate polyglot cloud applications in Cloud Foundry.	BC-CP-CF	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
SAP Excise Tax Management	Help Excise Tax customers calculate, track, and comply with excise duty tax requirements in real time.	LOD-ET-INT	Cloud Foundry	AWS	Europe (Frankfurt)	Yes	Available
Service Manager	The central registry for service brokers and platforms in SAP BTP.	BC-NEO-SVCMGR	Cloud Foundry	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Service Manager	The central registry for service brokers and platforms in SAP BTP.	BC-NEO-SVCMGR	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore	Yes	Available
Service Manager	The central registry for service brokers and platforms in SAP BTP.	BC-NEO-SVCMGR	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Big Data Services	Ready-to-use, fully managed Hadoop and Spark.	BC-NEO-BDS		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Neo	SAP	UAE (Dubai) Australia (Sydney) Brazil (São Paulo) Canada (Toronto) Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) Japan (Tokyo) KSA (Riyadh) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs)		Available
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	AWS	Australia (Sydney) Singapore South Korea (Seoul) Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Europe (Frankfurt) EU Access Japan (Tokyo) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	Azure	Australia (Sydney) Singapore Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	GCP	Europe (Frankfurt) US Central (IA)	Yes	Available
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Enhanced Disaster Recovery Service	Restore your cloud production environment with minimal data loss.	BC-NEO-DR	Neo	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Keystore Service	Manage cryptographic keys and certificates.	BC-NEO-SEC-CPG	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Canada (Toronto) KSA (Riyadh) UAE (Dubai)		Available
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	AWS	Australia (Sydney) Brazil (São Paulo) Japan (Tokyo) US East (VA) Europe (Frankfurt) Singapore Canada (Montreal) South Korea (Seoul) Europe (Frankfurt) EU Access	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo) US East (VA) Singapore Switzerland (Zurich) EU Access Brazil (São Paulo)	Yes	Available
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	GCP	Europe (Frankfurt) US Central (IA) India (Mumbai) Israel (Tel Aviv) KSA (Dammam) public sector Australia (Sydney)	Yes	Available
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Custom Domain Service	Configure and expose your application under your own domain.	BC-NEO-INFRA	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)		Available
Custom Domain Service	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Custom Domain Service	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Australia (Sydney) Singapore	Yes	Available
Custom Domain Service	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt)	Yes	Available
Custom Domain Service	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
Destination Service	Retrieve information about destinations in the Cloud Foundry environment.	BC-CP-DEST-CF	Cloud Foundry	AWS	Europe (Frankfurt) Europe (Frankfurt) EU Access Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore South Korea (Seoul) Canada (Montreal) US East (VA)	Yes	Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Destination Service	Retrieve information about destinations in the Cloud Foundry environment.	BC-CP-DEST-CF	Cloud Foundry	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore Australia (Sydney)	Yes	Available
Destination Service	Retrieve information about destinations in the Cloud Foundry environment.	BC-CP-DEST-CF	Cloud Foundry	GCP	US Central (IA) Europe (Frankfurt) Israel (Tel Aviv)	Yes	Available
Destination Service	Retrieve information about destinations in the Cloud Foundry environment.	BC-CP-DEST-CF	Cloud Foundry	Alibaba	China (Shanghai)**	Yes	Available
SAP Secure Login Service for SAP GUI	Get client certificates via Secure Login Client (SLC) to log on with SAP GUI.	BC-CP-CF-SEC-SLS		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
Agentry	Develop and run Agentry metadata-driven mobile applications.	MOB-CLD-AGS	Neo	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) Australia (Sydney) Brazil (São Paulo) Canada (Toronto) Japan (Tokyo) UAE (Dubai) KSA (Riyadh)		Available
Streaming Analytics	Process continuous streams of event data in real time and act on the results.	HAN-SDS	Neo	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) KSA (Riyadh) Europe (Amsterdam) Europe (Frankfurt) UAE (Dubai)		Available

Service Name	Short Description	Support Component	Environment	Infrastructure	Region	Available as Trial	Release Status
SAP API Business Hub	Discover, explore and test the APIs offered by SAP.	LOD-CHB		SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available

## Additional Components

### Tools

Name	Support Component
SAP BTP Cockpit	BC-CP-CPT BC-CP-CF-CPT [Cloud Foundry environment UI]
SAP BTP Command Line Interface	BC-CP-TOOLS-CLI
SAP BTP Console Client	BC-NEO-CMDTOOL
SAP UI Development Toolkit for HTML5 (SAPUI5)	CA-WDE
SDK for SAP BTP, Neo Environment	BC-NEO-SDK
SAP HANA Cockpit 2.0	HAN-CLS-CPT

### Software Logistics

Name	Support Component
Deployment (Neo Environment)	BC-NEO-DPL
Deployment (Cloud Foundry Environment)	BC-XS-SL-DS

### Other

Name	Support Component
Infrastructure (Neo environment)	BC-NEO-IT-NW
SAP Cloud Management Service	BC-NEO-CIS
SAP Usage Data Management Service	BC-NEO-MET-REP
SAP Audit Log	BC-NEO-AUDITLOG (Neo environment) BC-CP-CF-SEC-AUDITLG (Cloud Foundry environment)

Name	Support Component
SAP Streaming Analytics	HAN-SDS
SAP Cloud Application Programming Model (CAP)	BC-XS-CDX BC-XS-CDX-COR (Compiler and CDS Language) BC-XS-CDX-JAV (Java Runtime) BC-XS-CDX-NJS (Node.js Runtime) BC-XS-CDX-TLS (Tools, IDEs, Build, Deployment)

# 9 Glossary

## SAP BTP Terminology

### A-G

application	Software hosted on SAP BTP and used by business users to fulfill certain tasks. Applications are created by developers and might make use of services offered on the platform.
<a href="#">application router [page 409]</a>	The single point-of-entry for an application running in the Cloud Foundry environment on SAP BTP. The application router is used to serve static content, authenticate users, rewrite URLs, and forward or proxy requests to other micro services while propagating user information.
availability	The durability and operational performance without failure of a system or component for an agreed amount of time, which is defined in the contract.
availability zone	A physically separate location with its own power supply, network, and cooling. Within a region, it serves as an individual failure domain. If one of the availability zones fails, the survival of the region is ensured.
<a href="#">boosters [page 259]</a>	A set of guided interactive steps that enable you to select, configure, and consume services on SAP BTP to achieve a specific technical goal.
<a href="#">buildpack [page 49]</a>	In the Cloud Foundry environment, buildpacks provide framework and runtime support for apps.
<a href="#">business service [page 459]</a>	Platform services that enable, facilitate, or accelerate the development of business process components and elements of an application.
<a href="#">cockpit [page 121]</a>	The central point of entry to key information about your accounts and applications, and for managing all activities associated with your account.
connectivity	Provides a secure, reliable, and easy-to-consume access to business systems or remote services, running either on-premise or in the cloud.
Cloud Foundry CLI	The Cloud Foundry Command Line Interface (cf CLI) is used to deploy and manage your applications in the Cloud Foundry environment.
<a href="#">cloud connector</a>	Cloud Connector serves as the link between on-demand applications in SAP BTP and existing on-premise systems. It combines an easy setup with a clear configuration of the systems that are exposed to SAP BTP.

<a href="#">cloud management tools [page 129]</a>	Cloud management tools represent the group of technologies designed for managing SAP BTP.  Cloud management tools is a synonym for the internally used term Foundation.
disaster	An event declared by SAP when there is a loss of utilities and services, and uncertainty about whether they can be restored within a reasonable period of time. When a disaster is declared, a disaster recovery plan comes into action.
disaster recovery (DR)	A set of policies, tools, and procedures to protect applications by preserving and rapidly resuming their availability in case of a disaster.
durability	The ability of a system to permanently store data without loss or corruption.
<a href="#">enterprise account [page 84]</a>	An enterprise account is usually associated with one SAP customer or partner and is typically subject to charges. It groups together different subaccounts that an administrator makes available to users for deploying applications.
<a href="#">environment [page 47]</a>	Constitutes the SAP BTP actual Platform-as-a-Service offering that allows for the development and administration of applications. Each environment provides at least one application runtime, its own user and role management logic, and tools (for example, command line utility). Environments are integrated into the platform at the subaccount level.
failover	The automated or manually triggered process of switching from one system to another redundant system in case of an unexpected or planned downtime.
<a href="#">formation [page 2027]</a>	A logical grouping of SAP systems that can be extended in a single business scenario.
<a href="#">global account [page 94]</a>	The realization of a contract you made with SAP. A global account is region- and environment-independent, and it is used to manage subaccounts, members, entitlements and quotas.

## H-R

<a href="#">identity provider [page 3021]</a>	An authorization authority containing all user information and credentials. In SAP BTP, user information is provided by identity providers, not stored in SAP BTP itself.
<a href="#">member [page 104]</a>	Indicates a user's assignment to an account. As an account member, a user automatically has the permissions required to use the SAP BTP functionality within the scope of the respective account and as permitted by their account member roles.
<a href="#">OAuth</a>	Widely adopted security protocol for protection of resources over the Internet. It is used by many social network providers and by corporate networks.
<a href="#">org [page 104]</a>	A hierarchical level in the account structure of SAP BTP using a Cloud Foundry subaccount. Each Cloud Foundry subaccount contains exactly one Cloud Foundry org. Within an org, you can create several spaces.

platform service	Software that enables, facilitates, or accelerates the development of applications and other platform services on SAP BTP. Platform services are integrated with applications and other cloud resources by developers. End users only interact with platform services via applications, never directly. All platform services provide an interface such as an API or a set of APIs. There are two types of platform services: business and technical services.
programming model	A set of concepts used to create applications on SAP BTP. For example, a programming model can include programming languages, runtimes, and APIs.
<a href="#">quota [page 100]</a>	A numeric quantity that defines the maximum allowed consumption of a specific technical asset/resource.
<a href="#">region [page 17]</a>	A geographical location that usually consists of two or more availability zones.
resilience	The ability to provide and maintain an acceptable level of service in the face of faults and challenges until normal operation is restored.
resilient software design	Building and preparing an application or service in a way that it can handle failures that occur during runtime. Its goal is not to reduce the probability of failure occurrence, but to maximize the availability of systems and system landscapes in such cases.
runtime	An engine or context for executing programs, such as Java Web Tomcat 8 or Node.js runtime.

## S-Z

<a href="#">SAP BTP, Cloud Foundry environment [page 49]</a>	An open Platform-as-a-Service, which provides a scalable runtime container and a choice of clouds, runtimes, and services.
<a href="#">SAP BTP, Neo environment [page 81]</a>	An enterprise Platform-as-a-Service, providing a range of services to our customers
<a href="#">SAP BTP, Kyma runtime [page 70]</a>	A runtime the developers can use to build cloud-native Kubernetes-based extensions to SAP by using microservices and serverless Functions.
<a href="#">Pay-As-You-Go for SAP BTP [page 87]</a>	A variant to the consumption-based commercial model of SAP BTP in which customers of SAP BTP are billed and are paying for the exact services used. This differs from the CPEA payment model where customers purchase a committed amount of services in advance.
<a href="#">btp CLI [page 2321]</a>	The command line tool for all tasks on global account, directory, and subaccount level, such as creating or updating subaccounts, authorization management, and working with service brokers and platforms.
<a href="#">SAP ID service [page 3021]</a>	The default identity provider for SAP BTP applications. It manages the user base for SAP Community Network and other SAP Web sites. SAP ID service is also used for authentication in the cockpit and operations such as deploying, updating, and so on.
<a href="#">SAPUI5 [page 388]</a>	A development toolkit providing UI controls for developing Web applications.

<a href="#">service broker</a> [page 645]	When a developer provisions and binds a service to an application, the service broker for that service is responsible for providing the service instance and for binding services to applications. For example, the HANA service broker allows any application running on Cloud Foundry to connect to an SAP HANA database.
<a href="#">service plan</a> [page 100]	A variant of a service; for example, a database may be configured with various “t-shirt sizes”, each of which is a different service plan.
SAP Java Virtual Machine	SAP's own implementation of a Java Virtual Machine on which the SAP BTP infrastructure runs.
<a href="#">space</a> [page 2441]	In the Cloud Foundry environment, every application and service is scoped to a space. A space provides users with access to a shared location for application development, deployment, and maintenance. Each space role applies only to a particular space.
staging	The process in the Cloud Foundry environment by which the raw bits of an application are transformed into a droplet that is ready to execute.
<a href="#">subaccount</a> [page 94]	Lets you structure a global account according to customer requirements with regards to members, authorizations and quotas.  In the Cloud Foundry environment, a global account can have as many subaccounts as required; each subaccount can have an associated Cloud Foundry org.
<a href="#">subscription</a> [page 90]	One of the SAP BTP commercial models ("subscription-based"), where the customer subscribes to SAP BTP services for a certain amount of time and for a fixed price in order to get access to these services.
<a href="#">technical service</a> [page 16]	Platform services that enable, facilitate, or accelerate the generic development of an application, independent of the application's business process or task.
<a href="#">tool</a> [page 121]	A means for users to develop, configure, monitor and administer a service or entities managed by a service. A tool can be part of the platform or a service but is not a service by itself.
<a href="#">user-provided service instances</a> [page 252]	User-provided service instances enable you to use services that are not available in the marketplace with your applications running in the Cloud Foundry environment.

# Important Disclaimers and Legal Information

## Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon  : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon  : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

## Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

## Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.  
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

## Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

## Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.