



PUBLIC  
2020-09-10

# SAP Cloud Platform

# Content

<b>1</b>	<b>Working with SAP Cloud Platform .....</b>	<b>5</b>
<b>2</b>	<b>Getting Started.....</b>	<b>6</b>
2.1	Getting a Global Account.....	6
	Get a Trial Account.....	7
	Get an Enterprise Account.....	8
2.2	Getting Started in the Cloud Foundry Environment.....	9
	Getting Started with a Trial Account in the Cloud Foundry Environment.....	10
	Getting Started with an Enterprise Account in the Cloud Foundry Environment.....	21
	Getting Started with Multitenant Application Subscriptions in the Cloud Foundry Environment .....	29
2.3	Getting Started in the ABAP Environment.....	35
	Getting Started with a Trial Account in the ABAP Environment.....	35
	Getting Started with a Customer Account: Workflow in the ABAP Environment.....	40
2.4	Getting Started in the Kyma Environment.....	50
<b>3</b>	<b>Development.....</b>	<b>52</b>
3.1	Development in the Cloud Foundry Environment.....	53
	Developing Your First Application on SAP Cloud Platform.....	53
	Developing with the SAP Cloud Application Programming Model.....	54
	APIs in the Cloud Foundry Environment.....	59
	Business Application Pattern.....	60
	Deploy Business Applications in the Cloud Foundry Environment.....	62
	Deploy Docker Images in the Cloud Foundry Environment.....	63
	Developing SAP HANA in the Cloud Foundry Environment.....	64
	Developing HTML5 Applications in the Cloud Foundry Environment.....	66
	Developing Java in the Cloud Foundry Environment.....	146
	Developing Multitenant Applications in the Cloud Foundry Environment.....	194
	Developing Node.js in the Cloud Foundry Environment.....	224
	Developing Python in the Cloud Foundry Environment.....	237
	Developing SAPUI5.....	248
	Developing Security Artifacts.....	251
	Multitarget Applications in the Cloud Foundry Environment.....	286
	Using Services in the Cloud Foundry Environment.....	370
3.2	Development in the ABAP Environment.....	385
	ABAP Development User Guides.....	386
	ABAP Keyword Documentation.....	387

ABAP Lifecycle Management . . . . .	387
ABAP RESTful Application Programming Model . . . . .	432
Connect to the ABAP System . . . . .	433
Integration and Connectivity . . . . .	436
Released Components and Objects . . . . .	487
UI Development . . . . .	631
<b>3.3 Development in the Kyma Environment . . . . .</b>	<b>637</b>
Using Services in the Kyma Environment . . . . .	637
Create Functions . . . . .	646
<b>4 Extensions . . . . .</b>	<b>647</b>
<b>4.1 Extending SAP Solutions Using Automated Configurations [Feature Set A] . . . . .</b>	<b>648</b>
Registering an SAP System . . . . .	650
Assigning SAP Systems to a Formation . . . . .	652
Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment . . . . .	654
Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment . . . . .	692
Extending SAP Customer Experience Products in the Kyma Environment . . . . .	705
<b>4.2 Extending SAP Solutions Using Manual Configurations . . . . .</b>	<b>709</b>
Extending SAP S/4HANA Cloud in the Cloud Foundry Environment Manually . . . . .	709
Extending SAP SuccessFactors in the Cloud Foundry Environment Manually . . . . .	725
Extending SAP Cloud for Customer in the Cloud Foundry Environment Manually . . . . .	733
<b>5 Administration and Operations . . . . .</b>	<b>753</b>
<b>5.1 Account Administration . . . . .</b>	<b>753</b>
Account Administration in the Cockpit . . . . .	754
Account Administration Using the CLI for SAP Cloud Platform (sapcp CLI) [Feature Set B] . . . . .	850
Account Administration Using APIs (Beta) [Feature Set B] . . . . .	880
Manage Authentication and Authorization Using APIs . . . . .	911
<b>5.2 Administration and Operations in the Cloud Foundry Environment . . . . .</b>	<b>916</b>
Org Administration Using the Cockpit . . . . .	916
Org Administration Using the Cloud Foundry CLI . . . . .	930
Application Operations in the Cloud Foundry Environment . . . . .	979
Make Your Custom Identity Provider Visible in the Login Screen . . . . .	987
Audit Logging in the Cloud Foundry Environment . . . . .	988
<b>5.3 Administration and Operations in the ABAP Environment . . . . .</b>	<b>993</b>
Application Operations in the ABAP Environment . . . . .	993
<b>5.4 Administration and Operations in the Kyma Environment . . . . .</b>	<b>1079</b>
Roles in the Kyma Environment . . . . .	1080
Provision Kyma Environment . . . . .	1083
Kyma Environment Backup . . . . .	1083
<b>6 Security . . . . .</b>	<b>1085</b>

6.1	SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment . . . . .	1088
	What Is Authorization and Trust Management? . . . . .	1096
	Concepts of Authorization and Trust Management . . . . .	1125
	Terms and Definitions . . . . .	1128
	Monitoring and Troubleshooting . . . . .	1130
6.2	Default Role Collections of SAP Cloud Platform Cloud Foundry Environment [Feature Set B] . . . . .	1133
6.3	Principal Propagation . . . . .	1136
	Principal Propagation from the Neo to the Cloud Foundry Environment . . . . .	1136
	Principal Propagation from the Cloud Foundry to the Neo Environment . . . . .	1143
6.4	Data Protection and Privacy . . . . .	1151
	Glossary for Data Protection and Privacy . . . . .	1153
	Change Logging and Read-Access Logging . . . . .	1154
	Personal Data Record . . . . .	1155
	Deletion . . . . .	1155
	Consent . . . . .	1158
6.5	Security in the Kyma Environment . . . . .	1158
	Function Security . . . . .	1159
	Data Protection and Privacy in the Kyma Environment . . . . .	1159
<b>7</b>	<b>Getting Support . . . . .</b>	<b>1161</b>
1.	Check Platform Status . . . . .	1161
2.	Check Tools Versions . . . . .	1161
3.	Log on to SAP Support Portal . . . . .	1162
4.	Identify Affected System . . . . .	1162
5.	Provide Incident Details . . . . .	1162
7.1	Providing Details for Database Problems in the Cloud Foundry Environment . . . . .	1163
7.2	Gather Support Information . . . . .	1165
7.3	Platform Updates and Notifications in the Cloud Foundry Environment . . . . .	1165
7.4	Operating Model in the Cloud Foundry Environment . . . . .	1167
7.5	Support Components . . . . .	1174
<b>8</b>	<b>Glossary . . . . .</b>	<b>1284</b>

# 1 Working with SAP Cloud Platform

SAP Cloud Platform is an enterprise platform-as-a-service (enterprise PaaS) that provides comprehensive application development services and capabilities, which lets you build, extend, and integrate business applications in the cloud.

- **[Getting Started \[page 6\]](#)**

Take your first steps in the system. Follow the workflows for trial or customer accounts or subscribe to business applications.

- **[Development \[page 52\]](#)**

Develop and run business applications on SAP Cloud Platform. Use our application programming model, APIs, services, tools and capabilities.

- **[Extensions \[page 647\]](#)**

SAP Cloud Platform is the extension platform from SAP. It 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.

- **[Administration and Operations \[page 753\]](#)**

Manage and configure global accounts and subaccounts, and operate your applications in the different environments.

- **[Security \[page 1085\]](#)**

Use the security features and functions of SAP Cloud Platform to support the security policies of your organization.

- **[Getting Support \[page 1161\]](#)**

Use SAP Community, get guided answers, or explore SAP Support Portal.

## Related Information

[Basic Platform Concepts](#)

# 2 Getting Started

Get a global account and take your first steps in the system. Follow the workflows for trial or customer accounts or subscribe to business applications.

## [Getting a Global Account \[page 6\]](#)

SAP Cloud Platform offers two types of global accounts: Trial accounts and enterprise accounts.

## [Getting Started in the Cloud Foundry Environment \[page 9\]](#)

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

## [Getting Started in the ABAP Environment \[page 35\]](#)

Get onboarded in the ABAP environment of SAP Cloud Platform. Follow the workflows for trial or customer accounts.

## [Getting Started in the Kyma Environment \[page 50\]](#)

The getting started document describes the full list of steps you must complete as an administrator to set up a fully operational Kyma environment to which you can connect the chosen SAP solutions.

## Related Information

[Best Practices for SAP Cloud Platform](#)

## 2.1 Getting a Global Account

SAP Cloud Platform offers two types of global accounts: Trial accounts and enterprise accounts.

A **global account** is the realization of a contract you made with SAP. A global account is region-independent, and it 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 global accounts: enterprise accounts (paid) and trial accounts (free).

- [Get a Trial Account \[page 7\]](#)
- [Get an Enterprise Account \[page 8\]](#)

## 2.1.1 Get a Trial Account

Get a trial account to experience SAP Cloud Platform for free and gain access to a comprehensive set of platform services.

### Context

#### ! Restriction

There's currently no trial account available for SAP Cloud Platform in the China (Shanghai) region.

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 about the scope of our trial offering, see [Trial Accounts](#).

### Procedure

1. Go to the [SAP Cloud Platform website](#).
2. Register for a trial account by choosing *Start free trial*.
3. Create a new SAP ID by entering the required details or, if you have one already, log in.  
If you created a new SAP ID, you'll receive a confirmation email asking you to activate your account.
4. Choose the link in the e-mail to confirm your address and activate your trial account.
5. You are redirected to the SAP Cloud Platform website, where a dialog appears with your trial details. In that dialog, choose *Click here to start your trial!* to navigate to the cockpit.
6. To get started with the Cloud Foundry environment, simply choose *Enter Your Trial Account*.
7. Verify your phone number.

#### i Note

A phone number can only be used once. It is not possible to use the same phone number to verify multiple trial accounts.

If you don't receive an SMS with a verification code within 5 minutes, try again. If the problem persists, contact [sso@sap.com](mailto:sso@sap.com) with your account ID, e-mail address and phone number.

8. Once you've verified your phone number, log in to the cockpit again.
9. Select the region closest to you and choose *OK*.  
A global account, subaccount, org, and space are automatically created for you.
10. Choose *Go to Space*.

## 2.1.2 Get an Enterprise Account

To use an enterprise account, you can either purchase a customer account or join the partner program to purchase a partner account.

For more information about the scope of our enterprise offering, see [Enterprise Accounts](#).

### 2.1.2.1 Purchase 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 more information, see <https://cloudplatform.sap.com/pricing.html>. Contact us on [SAP Cloud Platform](#) or via an SAP sales representative.

In addition, 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.

After you have purchased your customer account, you will receive an e-mail with a link to the Home page of SAP Cloud Platform.

#### Getting a Customer Account in China

If you are located in China and want to buy a global account on SAP Cloud Platform, you need to contact an SAP sales representative: <https://www.sap.cn/registration/contact.html>. Note that there is currently no trial account available in China.

### 2.1.2.2 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?partnerstype=BLD&engagement=0002&build=1](https://partneredge.sap.com/content/partnerregistration/en_us/registration.html?partnerstype=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/partner/become/partneredge-build/plan-design.html>.

## Next Steps

- [Getting Started in the Cloud Foundry Environment \[page 9\]](#)
- [Getting Started in the ABAP Environment \[page 35\]](#)
- [Getting Started in the Kyma Environment \[page 50\]](#)

## 2.2 Getting Started in the Cloud Foundry Environment

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

### [Getting Started with a Trial Account in the Cloud Foundry Environment \[page 10\]](#)

Quickly get started with a trial account.

### [Getting Started with an Enterprise Account in the Cloud Foundry Environment \[page 21\]](#)

Quickly get started with an enterprise account.

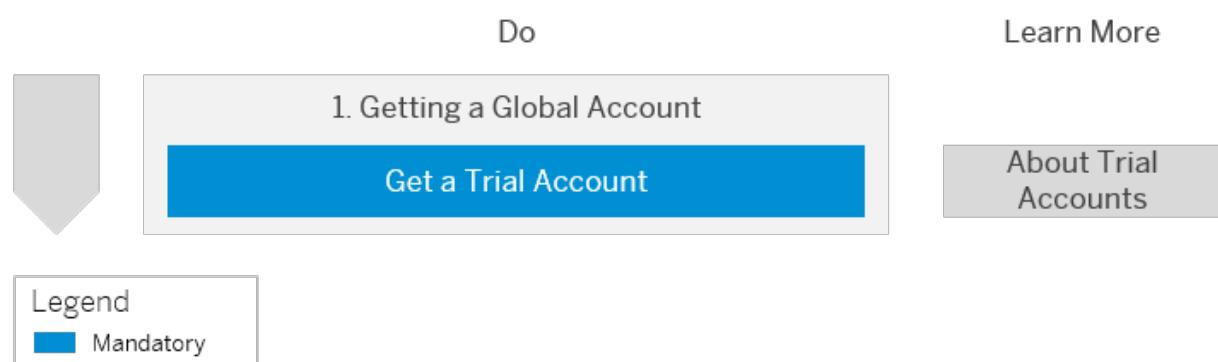
### [Getting Started with Multitenant Application Subscriptions in the Cloud Foundry Environment \[page 29\]](#)

As a global account owner on SAP Cloud Platform, you can develop and run applications in the Cloud Foundry environment, and also share them as multitenant applications with your own consumers. As a global account owner on SAP Cloud Platform, you can develop and run applications in the Cloud Foundry environment, and also share them as multitenant applications with your own consumers.

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

Quickly get started with a trial account.

### 1. Getting a Global Account

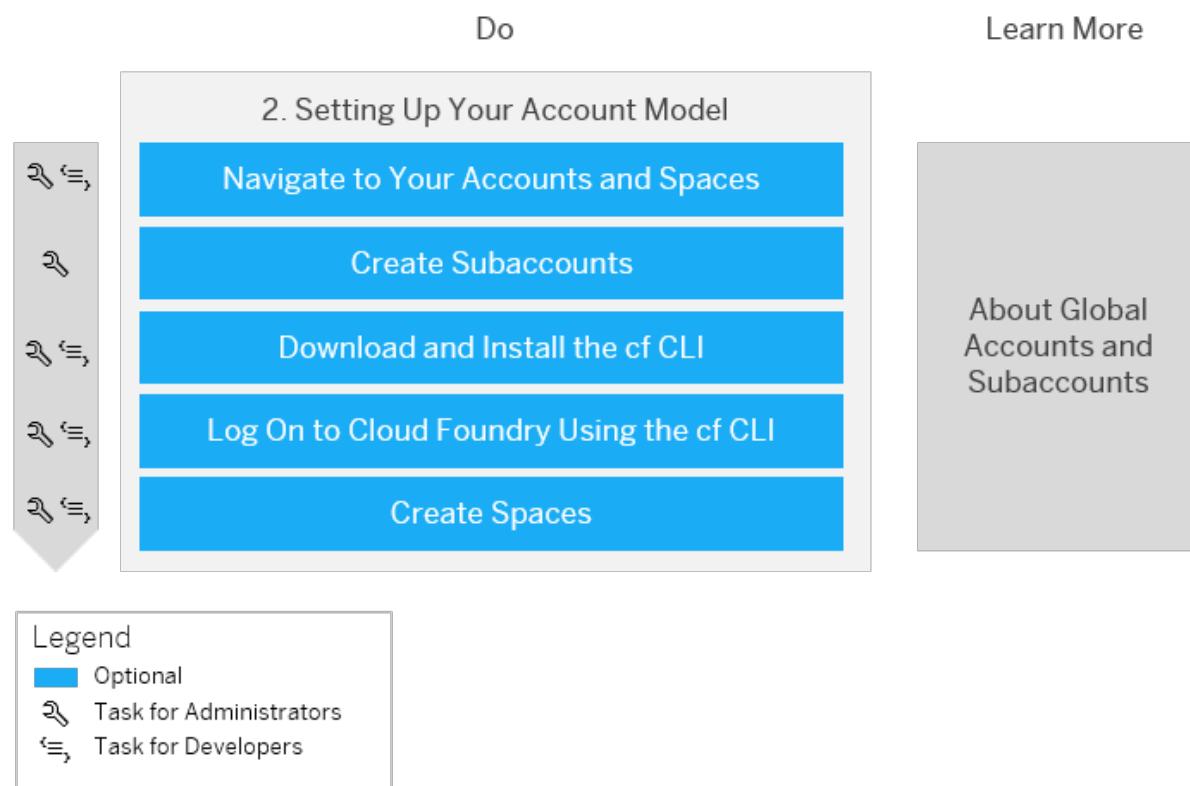


- [Get a Trial Account \[page 7\]](#)
- [Trial Accounts](#)

Before you begin, sign up for a free trial account. See [Get a Free Trial Account \[page 7\]](#). For more information about the scope of our trial offering, see [Trial Accounts](#).

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

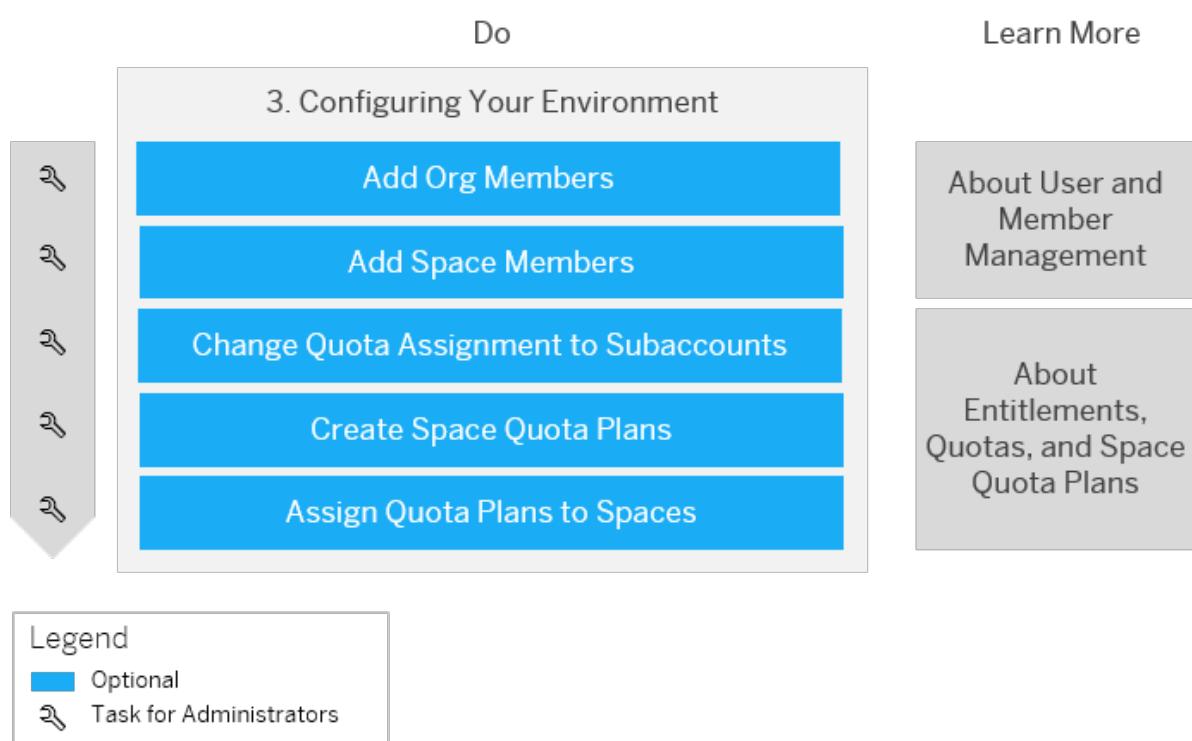
## 2. Setting Up Your Account Model



- [Navigate to Orgs and Spaces \[page 917\]](#)
  - [Create a Subaccount \[Feature Set A\] \[page 762\]](#)
  - [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#)
  - [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#)
  - [Create Spaces \[page 922\]](#)
  - [Relationship Between Global Accounts and Subaccounts \[Feature Set A\]](#)
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 cockpit. See [Navigate to Orgs and Spaces \[page 917\]](#).
  2. If you like, create further subaccounts. See [Create a Subaccount \[Feature Set B\] \[page 764\]](#). You can also download and use the CLI for SAP Cloud Platform to create new subaccounts. See [Download and Start Using the Client \[page 851\]](#) and [Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#).
  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 \[page 931\]](#). But don't worry, you can also perform all the necessary task using the SAP Cloud Platform 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 \[page 932\]](#).
- If you like, create further spaces. See [Create Spaces \[page 922\]](#). If you want to learn more about subaccounts, orgs, and spaces, and how they relate to each other, see [Account Model](#).

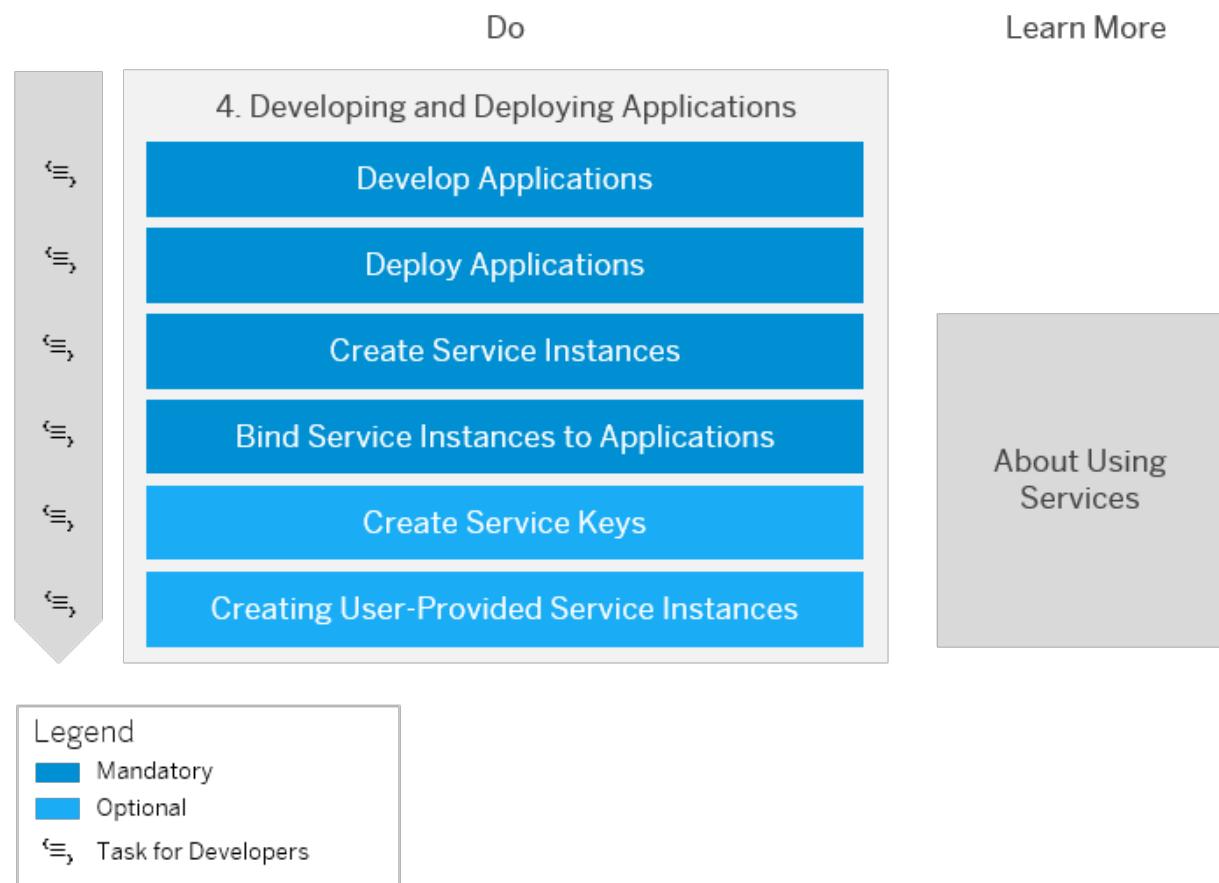
### 3. Configuring Your Environment



- Add Org Members Using the Cockpit [Feature Set A] [page 919]
  - Add Space Members Using the Cockpit [Feature Set A] [page 923]
  - Configure Entitlements and Quotas for Subaccounts [page 781]
  - Create Space Quota Plans [page 928]
  - Create Space Quota Plans Using the Cloud Foundry Command Line Interface [page 967]
  - User and Member Management
  - Entitlements and Quotas
- 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 Using the Cockpit \[Feature Set A\] \[page 919\]](#). For more information about the roles that are available on the different levels, see [User and Member Management](#).
  - You can also add members at the space level. See [Add Space Members Using the Cockpit \[Feature Set A\] \[page 923\]](#).

- In a trial account, quotas are automatically assigned to your subaccounts, but you can change that assignment. See [Configure Entitlements and Quotas for Subaccounts \[page 781\]](#). To learn more about entitlements and quotas, see [Entitlements and Quotas](#).
- You can also assign quotas to different spaces within a subaccount. To do so, first create a space quota plan. See [Create Space Quota Plans \[page 928\]](#) or [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#).
- Then assign the quota plan to your space. See [Assign Quota Plans to Spaces \[page 929\]](#) or [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 968\]](#).

## 4. Developing and Deploying Applications



- [Development \[page 52\]](#)
- [Deploy Business Applications in the Cloud Foundry Environment \[page 62\]](#)
- [Creating Service Instances \[page 372\]](#)
- [Binding Service Instances to Applications \[page 376\]](#)
- [Creating Service Keys \[page 379\]](#)

- [About Services \[page 371\]](#)
  - [Creating User-Provided Service Instances \[page 374\]](#)
1. Develop your application. Check out the Developer Guide for tutorials and more information. See [Development \[page 52\]](#).
  2. Deploy your application. See [Deploy Business Applications in the Cloud Foundry Environment \[page 62\]](#).
  3. Integrate your application with a service. To do so, first create a service instance. See [Creating Service Instances \[page 372\]](#).
  4. Bind the service instance to your application. See [Binding Service Instances to Applications \[page 376\]](#).
  5. Alternatively, you can also create and use service keys. See [Creating Service Keys \[page 379\]](#). For more information on using services and creating service keys, see [About Services \[page 371\]](#).
  6. You can also create instances of user-provided services. See [Creating User-Provided Service Instances \[page 374\]](#).

→ 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 Cloud Platform 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](#)

[Trial Accounts](#)

[Account Model](#)

[Entitlements and Quotas](#)

[User and Member Management](#)

[About Services \[page 371\]](#)

## 2.2.1.1 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.

### 2.2.1.1.1 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 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	<b>trial</b>
Description	Optional
Provider	Desired infrastructure provider
Region	Desired region
Subdomain	<b>&lt;your_id&gt;trial</b> Example: <b>P0123456789trial</b>
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.

## 2.2.1.1.2 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.

#### i 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.

## 2.2.1.1.3 Create Your Trial Space

### Procedure

1. In your *trial* subaccount, navigate to the *Spaces* page 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 Cloud Platform and the cockpit, as well as some more complex starter scenarios.

### 2.2.1.2 Setting Up a Trial Account via the Command Line [Feature Set B]

If your trial account is running on the cloud management tools feature set B, you can use the command-line interfaces to set it up. For all tasks on global account and subaccount level, you use the **CLI for SAP Cloud Platform** (in the following only referred to as **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 cockpit (except that you need the global account subdomain to log in, which you may have to look up in the cockpit).

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

For all tasks on global account and subaccount level, you can use the **CLI for SAP Cloud Platform** (in the following only referred to as **CLI**) instead of the cockpit. Once you've created a Cloud Foundry environment instance (a Cloud Foundry org), you use the Cloud Foundry CLI (cf CLI).

## Prerequisites

- You have created a trial account on cloud management tools feature set B.
- You have downloaded and extracted the following command-line tools:
  - CLI for SAP Cloud Platform (CLI). See [Download and Start Using the Client \[page 851\]](#)
  - Cloud Foundry CLI (cf CLI). See [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#)

#### → Tip

In the CLI for SAP Cloud Platform, 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 \[page 859\]](#).

Step No.	Task	Performed By	Do This	More Information
1.	Log in to your global account using the subdomain of your global account.	Global account administrator or viewer	Run this command in the CLI:  <code>sapcp login</code>	See <a href="#">Login [page 861]</a> .
2.	View details of your global account.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp get accounts/global-account</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI [page 869]</a> .
4.	View all the regions that are available to your global account and subaccounts.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp 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 account admin	Run this command in the CLI:  <code>sapcp 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 sapcp CLI [page 869]</a> and <a href="#">Relationship Between Global Accounts and Subaccounts [Feature Set A]</a> .
6.	View the details of the subaccounts in your global account.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp get accounts/subaccount &lt;ID of new subaccount&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI [page 869]</a> .
7.	Add admins to your subaccounts.	Subaccount admin	Assign the role collection Subaccount Administrator to the user by running the following command in the CLI:  <code>sapcp 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 sapcp CLI [page 873]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 784]</a> .
8.	View all the services and applications that are entitled to your global account, including quota information per service plan.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp list accounts/entitlement</code>	See <a href="#">Setting Entitlements Using the sapcp CLI [page 870]</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 CLI: <pre>sapcp 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>	See <a href="#">Setting Entitle-ments Using the sapcp CLI [page 870]</a> .
10.	View all the entitlements in your subaccounts.	Subac-count ad-min or viewer	Run this command in the CLI: <pre>sapcp list accounts/ entitlement --subaccount &lt;my-subaccount-id&gt;</pre>	See <a href="#">Setting Entitle-ments Using the sapcp CLI [page 870]</a> .
11.	Create a Cloud Foundry org (environment instance) in your subaccounts.	Subac-count ad-min	Run this command in the CLI: <pre>sapcp 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 sapcp CLI [page 871]</a> .
12.	View details of the environment instances in your subaccounts.	Subac-count ad-min or viewer	Run this command in the CLI: <pre>sapcp list accounts/ environment-instance -- subaccount &lt;my-subaccount- id&gt;</pre>	See <a href="#">Working with En-vironments Using the sapcp CLI [page 871]</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 965]</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 964]</a> and <a href="#">Add Space Members Using the Cloud Foundry Command Line Interface [page 966]</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 370]</a> .

Using the 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 CLI to work with multitenant applications. See [Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#).

## 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 \[page 53\]](#).

## Related Information

[Command Syntax \[page 855\]](#)

[Download and Start Using the Client \[page 851\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

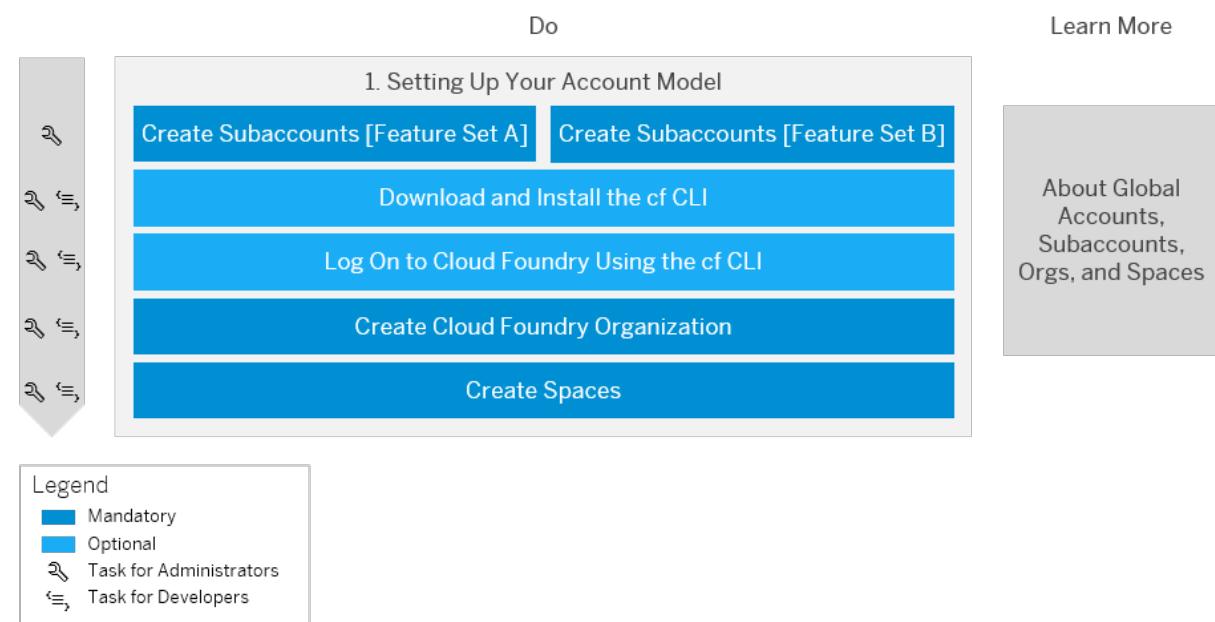
[Cloud Management Tools — Feature Set Overview](#)

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

Quickly get started with an enterprise account.

This topic focuses on how to get started with a customer or partner account using the SAP Cloud Platform cockpit. However, you can also perform these tasks using the CLI. See [Setting Up a Global Account via the Command Line \[Feature Set B\] \[page 25\]](#).

### 1. Setting Up Your Account Model

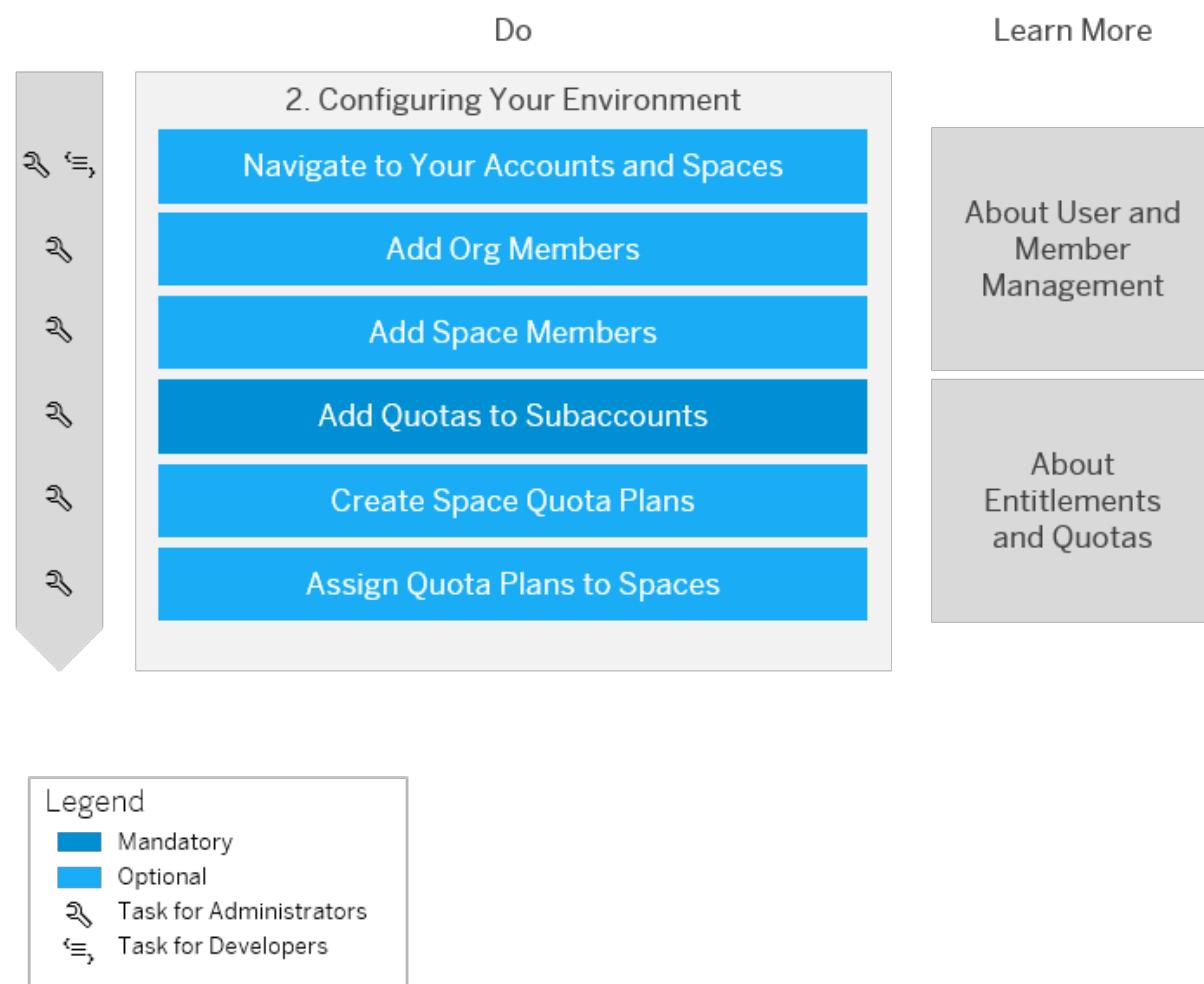


- Download and Install the Cloud Foundry Command Line Interface [page 931]
  - Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface [page 932]
  - Create Spaces [page 922]
  - Global Accounts
  - Create a Subaccount [Feature Set A] [page 762]
  - Create a Subaccount [Feature Set B] [page 764]
  - Create Orgs [page 918]
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 \[Feature Set A\] \[page 762\]](#) or [Create a Subaccount \[Feature Set B\] \[page 764\]](#).
  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 \[page 931\]](#).

But don't worry, you can also perform all the necessary task using the SAP Cloud Platform 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 \[page 932\]](#).
4. Create a Cloud Foundry organization in each of your subaccounts. See [Create Orgs \[page 918\]](#)
5. Create spaces. See [Create Spaces \[page 922\]](#).

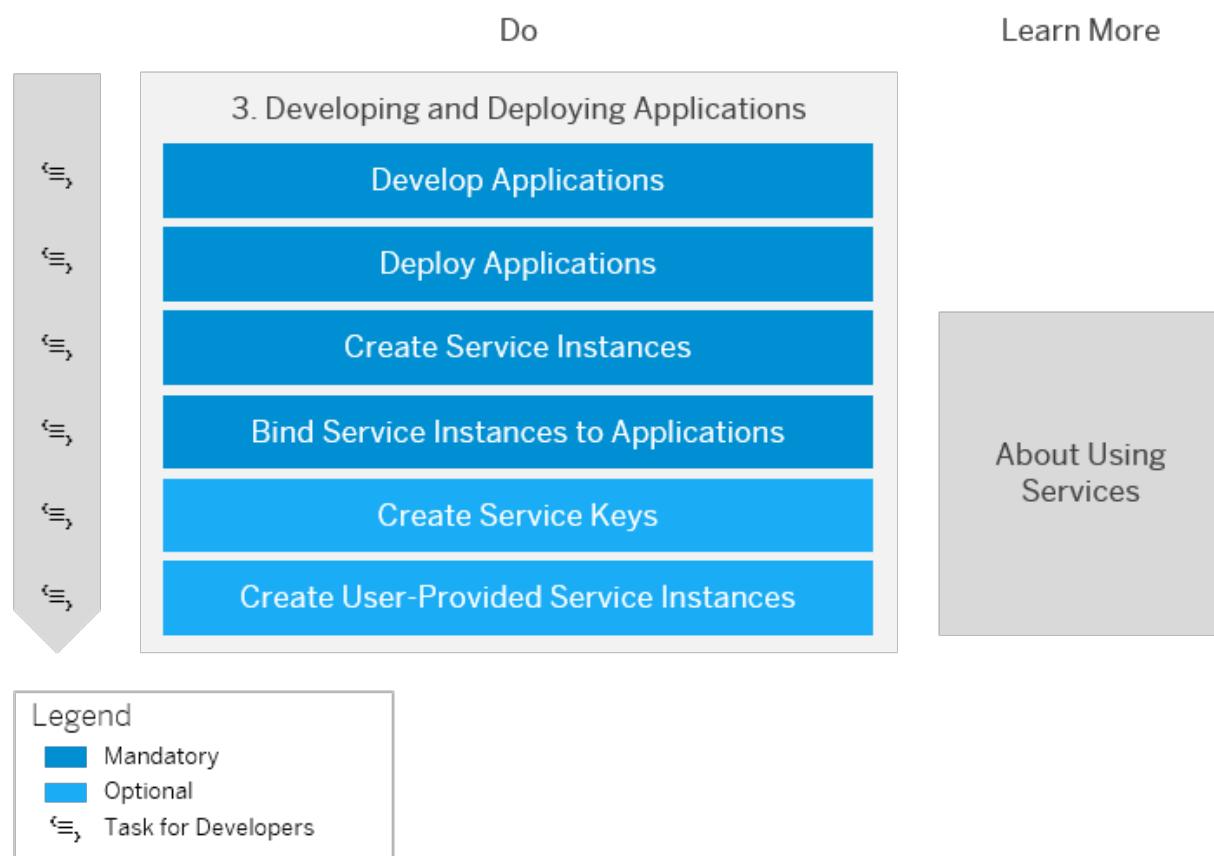
## 2. Configuring Your Environment



- User and Member Management
- Navigate to Orgs and Spaces [page 917]
- Add Org Members Using the Cockpit [Feature Set A] [page 919]
- Add Space Members Using the Cockpit [Feature Set A] [page 923]
- Configure Entitlements and Quotas for Subaccounts [page 781]

- [Create Space Quota Plans \[page 928\]](#)
  - [Assign Quota Plans to Spaces \[page 929\]](#)
  - [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#)
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 \[page 917\]](#).
  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 Using the Cockpit \[Feature Set A\] \[page 919\]](#). For more information about roles, see [User and Member Management](#).
  3. You can also add members at a space level. See [Add Space Members Using the Cockpit \[Feature Set A\] \[page 923\]](#).
  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 \[page 781\]](#). To learn more about entitlements and quotas, see [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).
  5. You can also assign quotas to different spaces within a subaccount. To do so, first create a space quota plan. See [Create Space Quota Plans \[page 928\]](#) or [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#).
  6. Then assign the quota plan to your space. See [Assign Quota Plans to Spaces \[page 929\]](#) or [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 968\]](#).

### 3. Developing and Deploying Applications



- [Creating Service Instances \[page 372\]](#)
  - [Binding Service Instances to Applications \[page 376\]](#)
  - [Creating Service Keys \[page 379\]](#)
  - [Using Services in the Cloud Foundry Environment \[page 370\]](#)
  - [Deploy Business Applications in the Cloud Foundry Environment \[page 62\]](#)
  - [Development \[page 52\]](#)
  - [Creating User-Provided Service Instances \[page 374\]](#)
1. Develop your application. Check out the Developer Guide for tutorials and more information. See [Development \[page 52\]](#).
  2. Deploy your application. See [Deploy Business Applications in the Cloud Foundry Environment \[page 62\]](#).
  3. Integrate your application with a service. To do so, first create a service instance. See [Creating Service Instances \[page 372\]](#)
  4. Bind the service instance to your application. See [Binding Service Instances to Applications \[page 376\]](#).
  5. Alternatively, you can also create and use service keys. See [Creating Service Keys \[page 379\]](#). For more information on using services and creating service keys, see [About Services \[page 371\]](#).
  6. You can also create instances of user-provided services. See [Creating User-Provided Service Instances \[page 374\]](#).

## Related Information

FAQ for Cloud Foundry environment within SAP Cloud Platform on the SAP Cloud Platform Public Wiki 

### 2.2.2.1 Setting Up a Global Account via the Command Line [Feature Set B]

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 Cloud Platform. You can use the command-line tool **CLI for SAP Cloud Platform** (in the following only referred to as **CLI**) to set it up.

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

Set up your account model by creating subaccounts in your enterprise account that is running on the cloud management tools feature set B. You can create any number of subaccounts in the Cloud Foundry environment and region.

## Prerequisites

- You have purchased an enterprise global account.
- You have downloaded and extracted the following command-line tools:
  - CLI for SAP Cloud Platform (CLI). See [Download and Start Using the Client \[page 851\]](#)
  - Cloud Foundry CLI (cf CLI). See [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#)
- The initial admin of the global account must make sure the following information has been obtained from the cloud operator or their account executive:
  - URL of the server of the CLI for SAP Cloud Platform.
  - Subdomain name of your global account.

## Procedure

#### → Tip

In the 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 \[page 859\]](#).

Step No.	Task	Per-formed By	Do This	More Information
1.	Log in to your global account using the URL of the CLI server and the subdomain of your global account.	Global account administrator or viewer	Run this command in the CLI:  <code>sapcp login</code>	See <a href="#">Login [page 861]</a> .
2.	View details of your global account.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp get accounts/global-account</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI [page 869]</a> .
3.	Add admins to your global account.	Global account admin	Assign the role collection Global Account Administrator to a user by running the following command in the CLI:  <code>sapcp 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 sapcp CLI [page 873]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 784]</a> .
4.	View all the regions that are available to your global account and subaccounts.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp 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 account admin	Run this command in the CLI:  <code>sapcp 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 sapcp CLI [page 869]</a> and <a href="#">Relationship Between Global Accounts and Subaccounts [Feature Set A]</a> .
6.	View the details of the subaccounts in your global account.	Global account admin or viewer	Run this command in the CLI:  <code>sapcp get accounts/subaccount &lt;ID of new subaccount&gt;</code>	See <a href="#">Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI [page 869]</a> .

Step No.	Task	Per-formed By	Do This	More Information
7.	Add admins to your subaccounts.	Subac-count ad-min	<p>Assign the role collection Subaccount Administrator to the user by running the following command in the CLI:</p> <pre>sapcp assign security/ role-collection "Subaccount Administrator" --to-user &lt;user&gt; --create- user-if-missing</pre>	See <a href="#">Managing Users and Their Authorizations Using the sapcp CLI [page 873]</a> and <a href="#">Security Administration: Managing Authentication and Authorization [page 784]</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	<p>Run this command in the CLI:</p> <pre>sapcp list accounts/ entitlement</pre>	See <a href="#">Setting Entitlements Using the sapcp CLI [page 870]</a> .
9.	Assign quotas to your subaccounts.	Global ac-count ad-min	<p>Run this command in the CLI:</p> <pre>sapcp 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>	See <a href="#">Setting Entitlements Using the sapcp CLI [page 870]</a> .
10.	View all the entitlements in your subaccounts.	Subac-count ad-min or viewer	<p>Run this command in the CLI:</p> <pre>sapcp list accounts/ entitlement --subaccount &lt;my-subaccount-id&gt;</pre>	See <a href="#">Setting Entitlements Using the sapcp CLI [page 870]</a> .
11.	Create a Cloud Foundry org (environment instance) in your subaccounts.	Subac-count ad-min	<p>Run this command in the CLI:</p> <pre>sapcp 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 Environments Using the sapcp CLI [page 871]</a> .

Step No.	Task	Performed By	Do This	More Information
12.	View details of the environment instances in your subaccounts.	Subaccount admin or viewer	Run this command in the CLI:  <pre>sapcp list accounts/ environment-instance -- subaccount &lt;my-subaccount-id&gt;</pre>	See <a href="#">Working with Environments Using the sapcp CLI [page 871]</a> .
13.	Create a Cloud Foundry space.	Org manager	Run these cf CLI commands:  <pre>cf login cf create-space</pre>	See <a href="#">Create Spaces Using the Cloud Foundry Command Line Interface [page 965]</a> .
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 964]</a> and <a href="#">Add Space Members Using the Cloud Foundry Command Line Interface [page 966]</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 370]</a> .

Using the 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 CLI to work with multitenant applications. See [Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#).

## 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 \[page 53\]](#).

## Related Information

[Command Syntax \[page 855\]](#)

[Download and Start Using the Client \[page 851\]](#)  
[Commands in the sapcp CLI \[page 867\]](#)  
[Cloud Management Tools — Feature Set Overview](#)

## 2.2.3 Getting Started with Multitenant Application Subscriptions in the Cloud Foundry Environment

As a global account owner on SAP Cloud Platform, you can develop and run applications in the Cloud Foundry environment, and also share them as multitenant applications with your own consumers. As a global account owner on SAP Cloud Platform, you can develop and run applications in the Cloud Foundry environment, and also share them as multitenant applications with your own consumers.

1. [Providing Multitenant Applications to Consumers in the Cloud Foundry Environment \[page 29\]](#)
2. [Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit \[page 32\]](#)
3. [Configure Application Roles and Assign Roles to Users \[page 34\]](#)

### Related Information

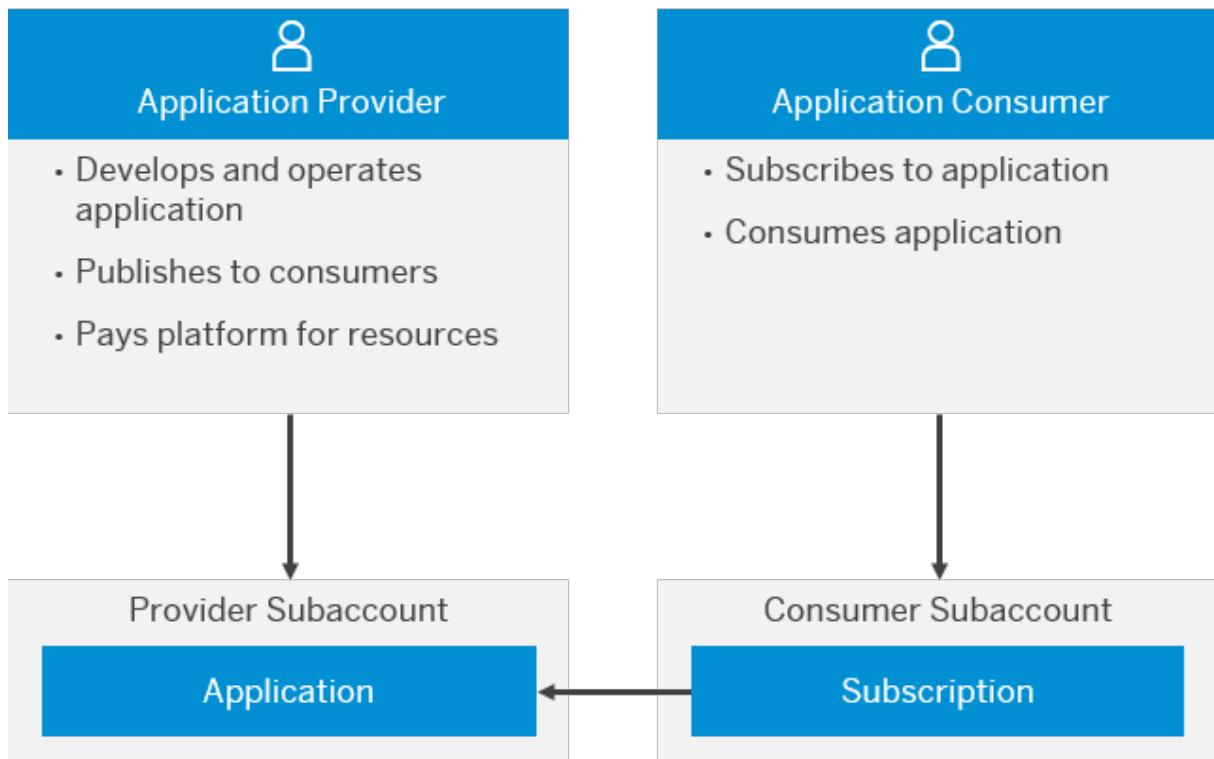
[Developing Multitenant Applications in the Cloud Foundry Environment \[page 194\]](#)

### 2.2.3.1 Providing Multitenant Applications to Consumers in the Cloud Foundry Environment

Software developers can use SAP Cloud Platform to build and run applications, and then share them with multiple consumers, such as units in your organization. These shared or reused applications are called **multitenant applications**.

### What is Multitenancy?

SAP Cloud Platform provides a multitenant functionality that allows application providers to own, deploy, and operate applications for multiple consumers, with reduced costs. For example, the provider can upgrade the application for all consumers instead of performing each update individually, or can share resources across multiple consumers. Application consumers launch the applications using consumer-specific URLs, and can configure certain application features.



Consumers cannot see the data of other consumers; the multitenant application maintains data separation between tenants, and the identity access management (IAM) is kept isolated. Each consumer accesses the multitenant application through a dedicated URL.

Consumers subscribe to the provider application from their dedicated subaccount. This subscription represents the contract or relation between a consumer's subaccount and a provider application.

As with any application running in SAP Cloud Platform, 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.

## Multitenancy Roles

The multitenancy concept involves two main user roles:

Application provider	An SAP global account owner that uses SAP Cloud Platform 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.

## Consumer Subaccounts and Subaccount Members

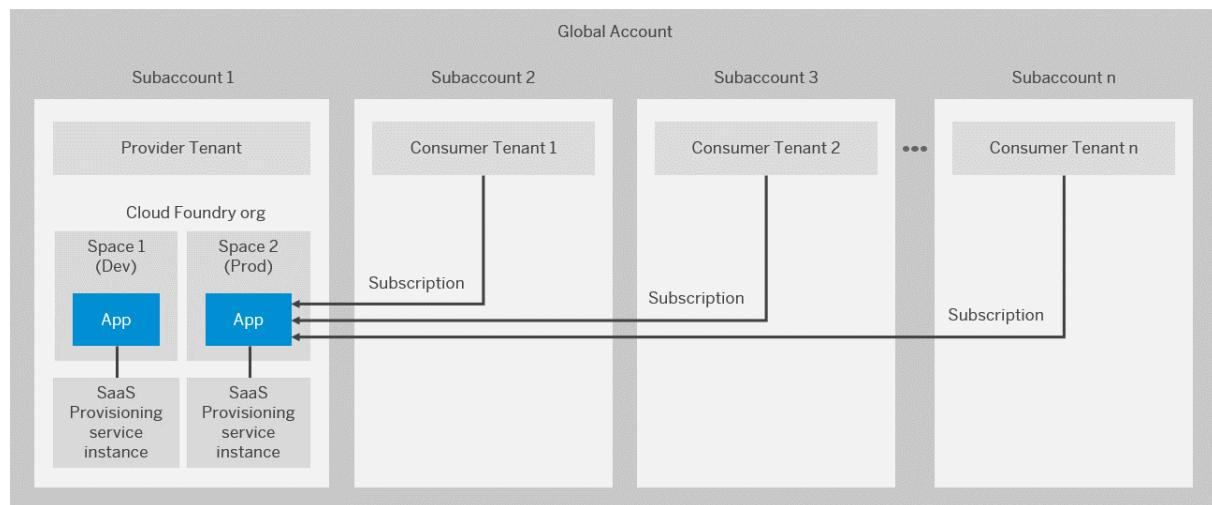
To use multitenancy in the Cloud Foundry environment in SAP Cloud Platform, the application provider must create a subaccount for each consumer in the same global account where the multitenant application is deployed.

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 cockpit.

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?

1. Develop, deploy, configure, and register your multitenant application.  
See [Developing Multitenant Applications in the Cloud Foundry Environment \[page 194\]](#).
2. Create a subaccount in your global account for each application consumer.  
See [Create a Subaccount \[Feature Set A\] \[page 762\]](#).

3. Subscribe each consumer subaccount to the hosted application deployed in the provider account.  
See [Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit \[page 32\]](#).
4. Set up application roles and assign users.  
See [Configure Application Roles and Assign Roles to Users \[page 34\]](#).

### 2.2.3.2 Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit

Subscribe to multitenant applications from the [Subscriptions](#) page in the SAP Cloud Platform cockpit.

#### Prerequisites

- If you are 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 Cloud Platform](#) or via an SAP sales representative.
- If you are an application owner who is subscribing consumer tenants to your own multitenant application:
  - The application must be deployed to your provider subaccount, configured, and registered as described in [Developing Multitenant Applications in the Cloud Foundry Environment \[page 194\]](#).
  - You have created a 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 \[page 29\]](#).
- [Feature Set A] You are an administrator of the global account.
- [Feature Set B] You are an administrator of the subaccount.
- [Feature Set B] Your subaccount is entitled to the multitenant application. See [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).

#### i Note

If you have cloud credits, then you are eligible to subscribe to any multitenant application that is available to your global account in the Cloud Foundry environment.

#### 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 to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#) or [Navigate to Global Accounts and Subaccounts \[Feature Set B\] \[page 757\]](#).

3. In the navigation area of each subaccount, choose *Subscriptions*.

The following information is displayed for the multitenant applications to which your global account is entitled in the Cloud Foundry environment:

- The name and short description of the application.
  - *Subscribed / Not subscribed*: The status of the application, indicating whether the subscription is active in your subaccount in the current region.
4. Click the application name to open its *Overview* page.
  5. Choose *Subscribe*.

The [Go to Application](#) link becomes available once the subscription is activated. Choose the link to launch the application and obtain its URL.

### i Note

To remove a subscribed application, go back to the *Overview* page of the subscribed application and then choose *Unsubscribe*. All data related to the multitenant application will be deleted in the respective subaccount.

## Next Steps

1. Configure user access to the application.  
See [Configure Application Roles and Assign Roles to Users \[page 34\]](#).
2. If the application consumer is not 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 sapcp CLI \[page 872\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 2.2.3.3 Configure Application Roles and Assign Roles to Users

View, create, and modify application roles and then assign users to these roles using the SAP Cloud Platform cockpit.

### Prerequisites

Your subaccount is subscribed to a multitenant application in the Cloud Foundry environment. See [Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit \[page 32\]](#).

### Context

You can use any SAML 2.0 standard compliant identity provider. See [Trust and Federation with Identity Providers \[page 789\]](#).

## View, Create, and Modify Application Roles

### Procedure

1. Navigate to your subaccount. For more information, see [Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#).
2. In the navigation area, choose *Subscriptions*.
3. Click the application name to open its *Overview* page.
4. Choose *Manage Roles* to view, create, and modify the application roles.

## Assign Users to Application Roles

### Procedure

1. Navigate to your subaccount. For more information, see [Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#).
2. Choose *Security* *Roles Collections* in the navigation area and include the roles into the roles collection.
3. Choose *Security* *Trust Configuration* and assign role collections to users.

## Related Information

[User and Member Management](#)

## 2.3 Getting Started in the ABAP Environment

Get onboarded in the ABAP environment of SAP Cloud Platform. Follow the workflows for trial or customer accounts.

[Getting Started with a Trial Account in the ABAP Environment \[page 35\]](#)

Quickly get started with a trial account.

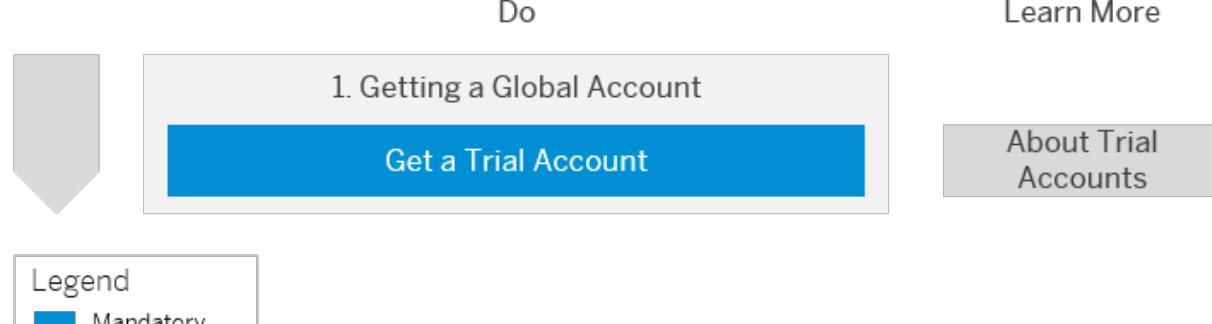
[Getting Started with a Customer Account: Workflow in the ABAP Environment \[page 40\]](#)

Quickly get started with a customer account.

### 2.3.1 Getting Started with a Trial Account in the ABAP Environment

Quickly get started with a trial account.

#### 1. Getting a Global Account



- [Get a Trial Account \[page 7\]](#)
- [Trial Accounts](#)

Before you begin, sign up for a free trial account. See [Get a Free Trial Account \[page 7\]](#). For more information about trial accounts, see [Trial Accounts](#).

### **! Restriction**

Note that you can only select [Amazon Web Services \(AWS\)](#) as your provider and either [Europe \(Frankfurt\)](#) or [US East \(VA\)](#) as your region to get access to ABAP trial.

## **2. Navigating to Your ABAP Trial Service**

Do

### **2. Navigating to the ABAP Trial Service**

**Navigate to Orgs and Spaces**

**Select Your ABAP Trial Service**

#### **Legend**

- Mandatory
- Optional

- [Navigate to Orgs and Spaces \[page 917\]](#)

1. After registering for a trial account, you will be navigated to the space that was created for your Cloud Foundry trial account. If not, see [Navigate to Orgs and Spaces \[page 917\]](#).

#### **i Note**

You should see the following path in the breadcrumbs:

Home /  <global\_account> /  <subaccount> /  <space>

2. Go to your trial service by selecting [ABAP Trial](#) from the list of services that are available to you in the [Service Marketplace](#).

#### **→ Tip**

If you don't see the [ABAP Trial](#) tile, go back to your trial subaccount and select [Entitlements](#) in the navigation menu to add a shared service plan. See [Configure SAP Cloud Platform Entitlements](#).

### 3. Setting Up Your ABAP Trial Instance

Do

#### 3. Setting Up Your Trial Instance

Create Your ABAP Trial Instance

##### Legend

 Mandatory

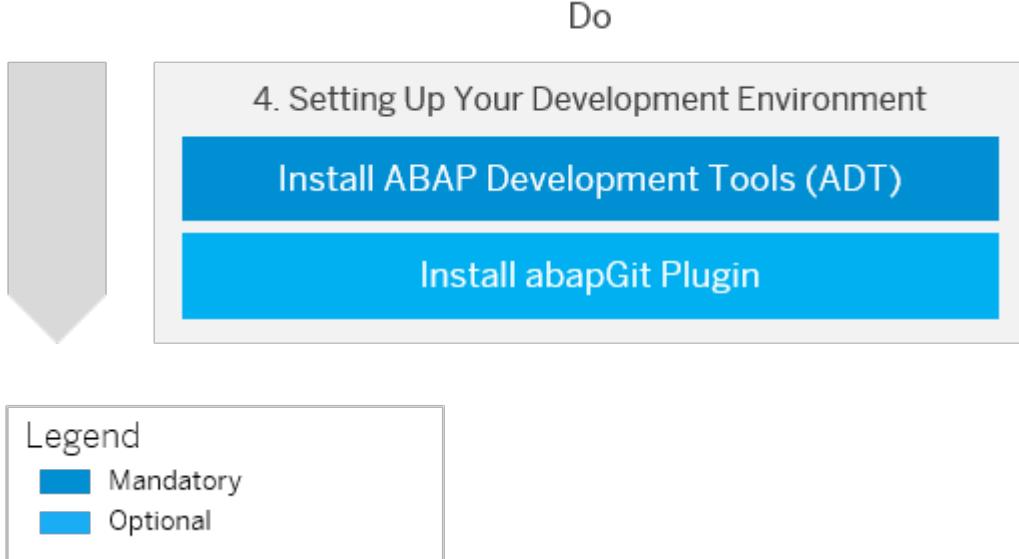
- [Create Your ABAP Trial Instance \[page 40\]](#)

Now that you have registered for a trial account and navigated to your ABAP trial service, it's time to set up your ABAP trial instance. See [Create Your ABAP Trial Instance \[page 40\]](#).

##### → Tip

If you experience issues with accessing your ABAP trial service or creating an ABAP trial instance, submit your question on [SAP Community](#) by selecting **SAP Cloud Platform, abap environment** as your primary tag.

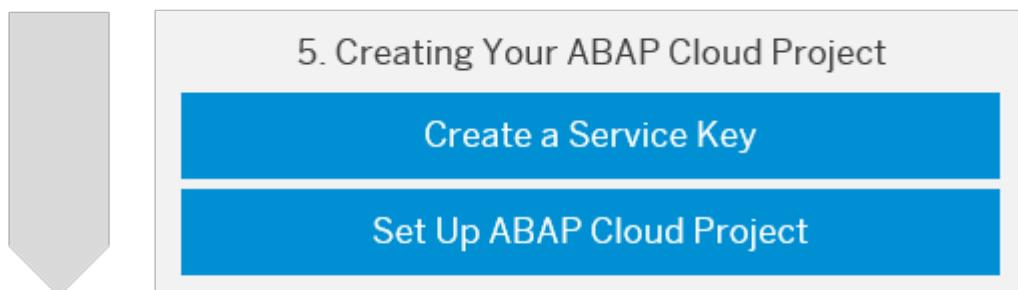
## 4. Setting Up Your Development Environment



- <https://tools.hana.ondemand.com/#abap> [https://tools.hana.ondemand.com/#abap]
  - <https://developers.sap.com/tutorials/abap-environment-abapgit.html#63bab1ab-0d66-4188-a693-8f63a2944d49> [https://developers.sap.com/tutorials/abap-environment-abapgit.html#63bab1ab-0d66-4188-a693-8f63a2944d49]
1. Download and install ABAP Development Tools (ADT) from <https://tools.hana.ondemand.com/#abap>.
  2. Install the abapGit plugin. See [Install and Set Up abapGit \[page 390\]](#) and [Tutorial: Install abapGit Eclipse plugin](#).

## 5. Creating Your ABAP Cloud Project

Do



### Legend

Mandatory

- [Creating Service Keys \[page 379\]](#)
  - [Connect to the ABAP System \[page 433\]](#)
1. Create a service key for your trial system. See [Creating Service Keys \[page 379\]](#).
  2. To start developing in your trial system, you need to create a new project in your ADT installation. Set up an ABAP cloud project to connect to your ABAP trial system. See [Connect to the ABAP System \[page 433\]](#).

**Parent topic:** [Getting Started in the ABAP Environment \[page 35\]](#)

## Related Information

[Getting Started with a Customer Account: Workflow in the ABAP Environment \[page 40\]](#)

[Video Tutorial: Getting Started with a Trial Account](#)

[Tutorial: Create an SAP Cloud Platform ABAP Environment Trial User](#)

[Tutorial: Create Your First ABAP Console Application](#)

[Trial Learning Journey](#)

[Learning Journey](#)

[Product Page](#)

[SAP Community](#)

[Solution Overview and Roadmap](#)

## 2.3.1.1 Create Your ABAP Trial Instance

Learn how to create your ABAP trial instance.

### Procedure

1. In the navigation area, select *Instances*.
2. Select *New Instance*.
3. Choose the *shared* plan and select *Next*.
4. Provide the email address you have used for setting up your trial account as a value for the *email* parameter in the JSON template and select *Next*.

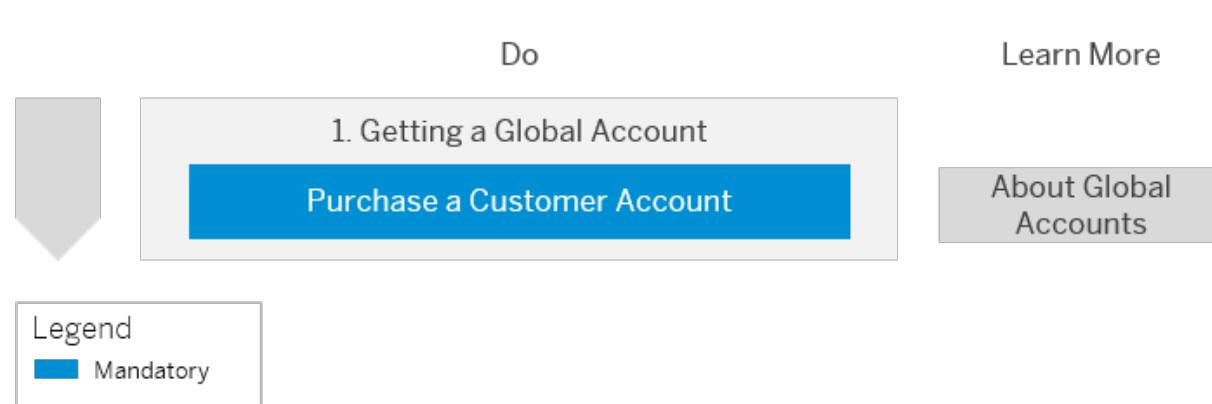
```
{  
  "email": "<your trial user's mail address>"  
}
```

5. [Optional] Choose an application to bind the new service instance to. See [Binding Service Instances to Applications \[page 376\]](#).
6. Select *Next*.
7. Choose *Instance Name* (e.g. abap-trial) and select *Finish*.

## 2.3.2 Getting Started with a Customer Account: Workflow in the ABAP Environment

Quickly get started with a customer account.

### 1. Getting a Global Account



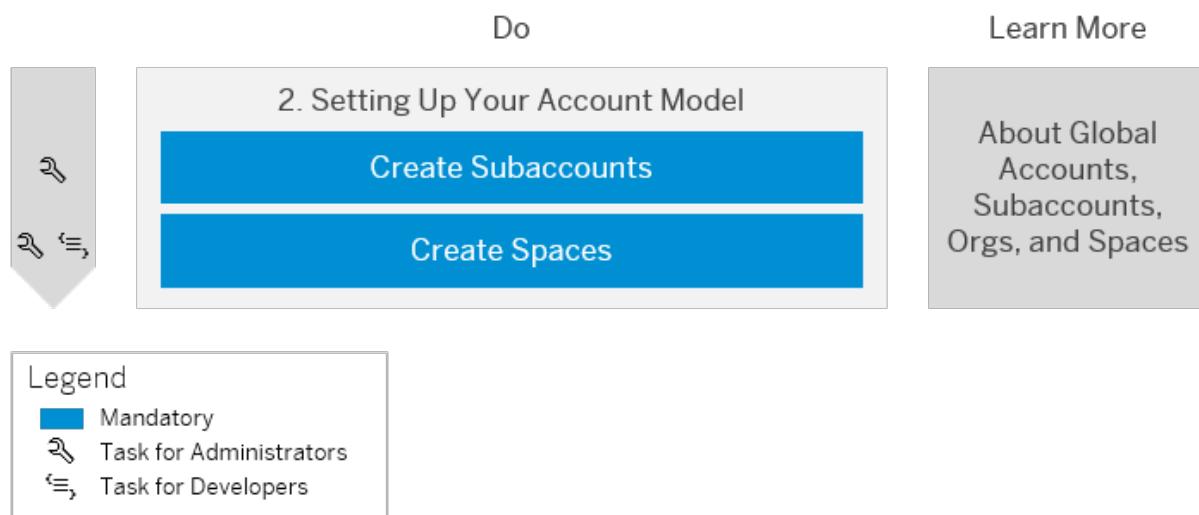
- Purchase a Customer Account [page 8]
- Trial Accounts

Before you begin, purchase a customer account. See [Purchase a Customer Account \[page 8\]](#).

### → Tip

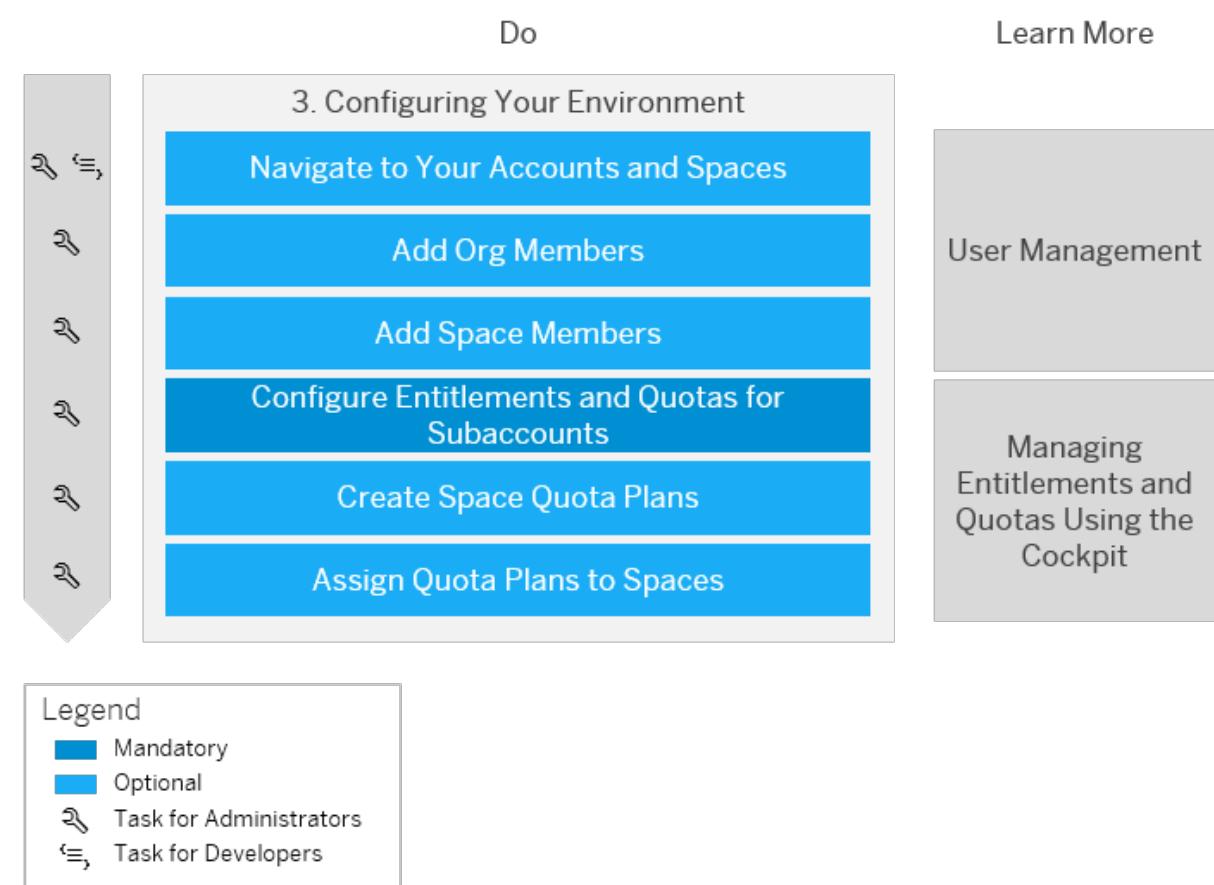
To speed up the setup process, log on to the cockpit and choose *Booster* in the navigation area. The interactive booster guides you through the process of setting up your subaccounts, configuring entitlements, and assigning members. This will give you a jumpstart into your ABAP development projects. See [Using a Booster to Automate the Setup of the ABAP Environment \[page 47\]](#). To find out more about boosters, see [Boosters \[page 384\]](#)

## 2. Setting Up Your Account Model



- Create a Subaccount [Feature Set A] [page 762]
  - Create Spaces [page 922]
  - Global Accounts
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 \[Feature Set A\] \[page 762\]](#).
  2. Create spaces. See [Create Spaces \[page 922\]](#). If you want to learn more about subaccounts, orgs, and spaces, and how they relate to each other, see [Account Model](#). You'll also find some recommendations for setting up your account model so that it meets your business needs.

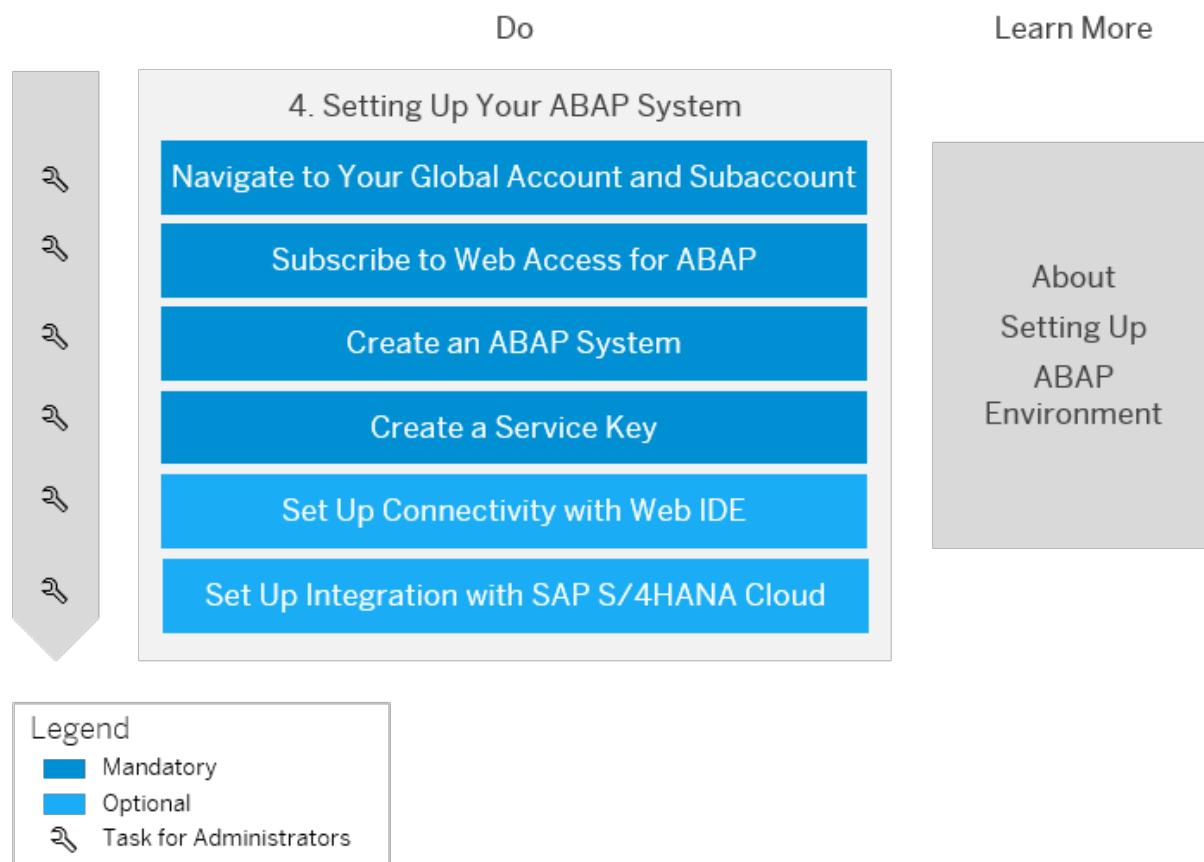
### 3. Configuring Your Environment



- [Navigate to Orgs and Spaces \[page 917\]](#)
  - [Assign Quota Plans to Spaces \[page 929\]](#)
  - [Create Space Quota Plans \[page 928\]](#)
  - [Add Members to Your Global Account \[Feature Set A\] \[page 760\]](#)
  - [Add Space Members Using the Cockpit \[Feature Set A\] \[page 923\]](#)
  - [Configure Entitlements and Quotas for Subaccounts \[page 781\]](#)
  - [User and Member Management](#)
  - [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#)
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 \[page 917\]](#).
  2. It's time to think about member management. You can add members at different levels. For example, you can add members at the global account level. See [Add Members to Your Global Account \[Feature Set A\] \[page 760\]](#). For more information about roles, see [User and Member Management](#).
  3. You can also add members at an org level. See [Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#),

4. You can also add members at a space level. See [Add Space Members Using the Cockpit \[Feature Set A\] \[page 923\]](#).
5. 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 \[page 781\]](#). To learn more about entitlements and quotas, see [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).
6. You can also assign quotas to different spaces within a subaccount. To do so, first create a space quota plan. See [Create Space Quota Plans \[page 928\]](#) or [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#).
7. Then assign the quota plan to your space. See [Assign Quota Plans to Spaces \[page 929\]](#) or [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 968\]](#).

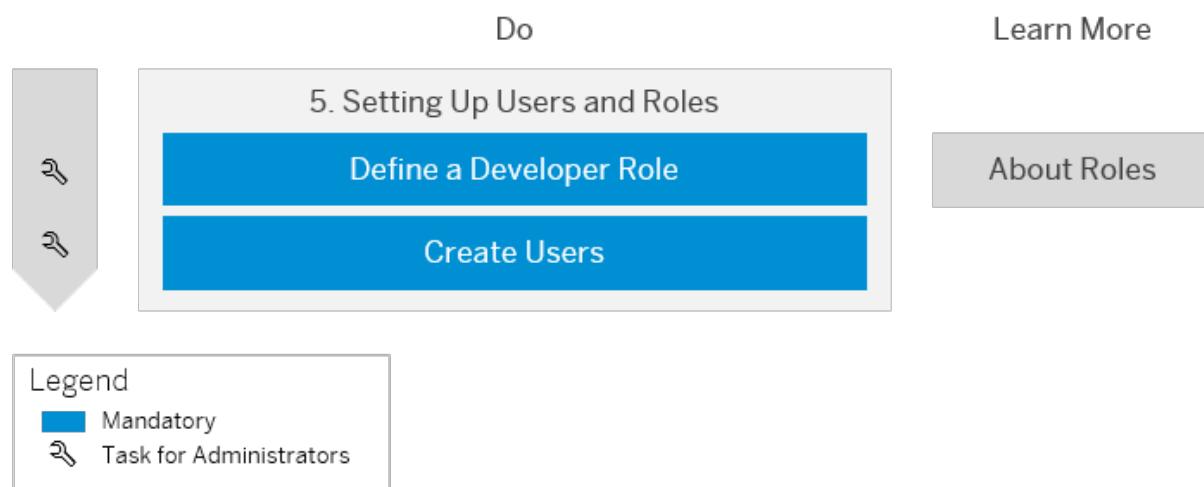
## 4. Setting Up Your ABAP System



- Setup of the ABAP Environment: Introduction
- Navigate to Global Accounts and Subaccounts [Feature Set A] [page 756]
- Subscribing to the Web Access for ABAP

- [Create an ABAP System \[page 48\]](#)
  - [Creating Service Keys \[page 379\]](#)
  - [Setup of the ABAP Environment: Introduction](#)
  - [Integration of SAP Cloud Platform ABAP Environment with SAP S/4HANA Cloud: Introduction](#)
1. Since you need to use the SAP Cloud Platform cockpit to configure your environment, it's important you understand how you can navigate to your global account and subaccounts. See [Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#).
  2. Subscribe to the Web access for ABAP SaaS application to get direct browser access to your instances in the ABAP environment. This also allows you to access the administration launchpad including your own SAP Fiori applications. You only have to subscribe once for each subaccount, which includes all systems to be created in all spaces of this subaccount. For further information, see [Subscribing to the Web Access for ABAP](#).
  3. Now it's time to create your ABAP system. To do so, follow the steps described in [Create an ABAP System \[page 48\]](#).
  4. Create a service key for the ABAP system that you've just created. Choose a meaningful name such as ADT for later access to ABAP Development Tools (ADT). For more information about service keys, see [Creating Service Keys \[page 379\]](#).
  5. You can set up your ABAP environment to ensure secure communication between the Web IDE in the Neo environment and Cloud Foundry environment. See [Setup of the ABAP Environment: Introduction](#).
  6. Integrate your ABAP environment with SAP S/4HANA Cloud. See [Integration of SAP Cloud Platform ABAP Environment with SAP S/4HANA Cloud: Introduction](#).

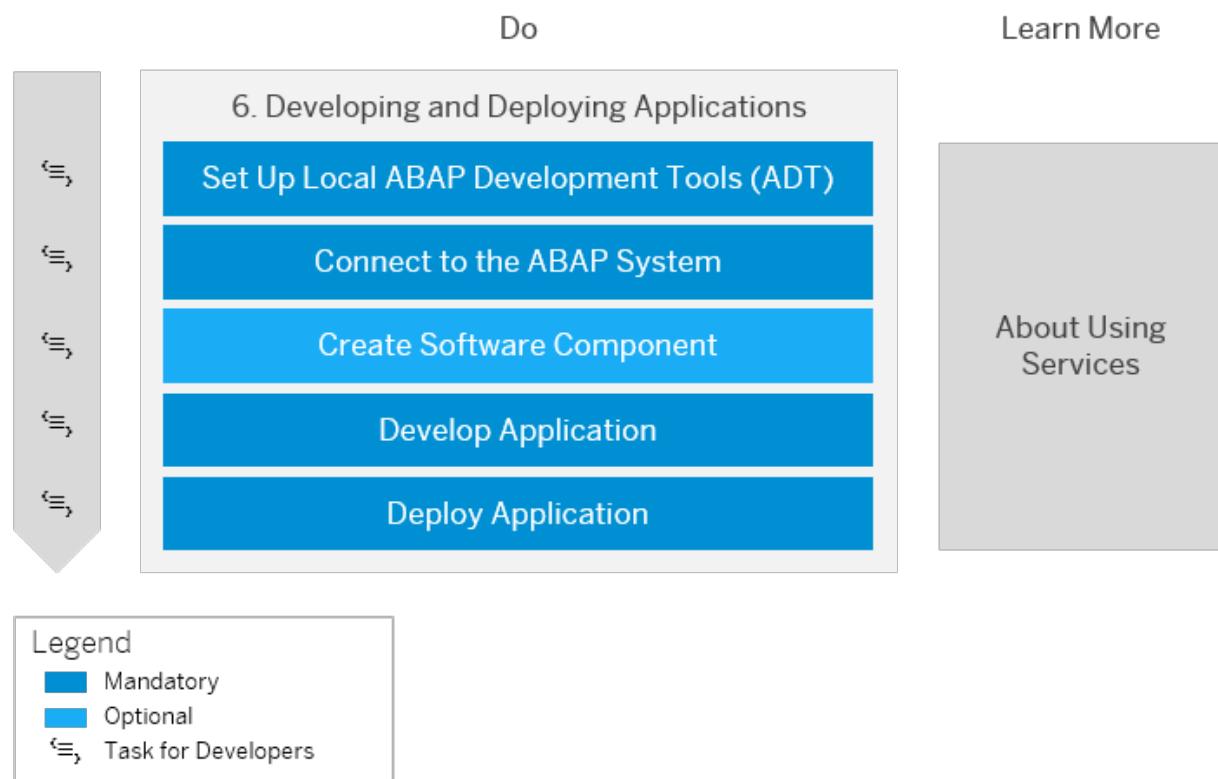
## 5. Setting Up Users and Roles



- [Identity and Access Management \[page 1038\]](#)
- [Define a Developer Role \[page 49\]](#)
- [Creating an Employee Record for the Developer in the ABAP Environment](#)

1. Define a developer business role with unrestricted write access. To do so, use a template that includes authorization to maintain all supported development entities such as CDS views, ABAP classes, etc. The created ABAP system provides the predefined business role template SAP\_BR\_DEVELOPER. To define your developer role including maintenance authorization, use this template and set the property for write access. See [Define a Developer Role \[page 49\]](#). For more information about creating and maintaining business roles, see [How to Create a Business Role from a Template \[page 1045\]](#) and [Maintain Business Roles \[page 1043\]](#).
2. Create business users and assign them developer business roles. See [Creating an Employee Record for the Developer in the ABAP Environment](#). For more information about managing business users, see [Maintain Employees \[page 1036\]](#).

## 6. Developing and Deploying Applications



- <https://tools.hana.ondemand.com/#abap> [<https://tools.hana.ondemand.com/#abap>]
- [Connect to the ABAP System \[page 433\]](#)
- [Development in the ABAP Environment \[page 385\]](#)
- [Using Services in the Cloud Foundry Environment \[page 370\]](#)
- [Software Component Lifecycle Management \[page 1068\]](#)

- <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]
1. Download and install ABAP Development Tools (ADT) from <https://tools.hana.ondemand.com/#abap>. See [Video Tutorial: Configure ABAP Development Tools](#).

### i Note

SAP GUI is no longer supported. 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 433].
3. Create a software component. See [Tutorial: Create Software Component via SAP Fiori Launchpad](#).
4. Develop your application. See [Development in the ABAP Environment](#) [page 385]. To learn more about how to develop applications, see [Tutorial: Create Your First ABAP Console Application](#) and [Video Tutorial: Create Application](#).
5. Deploy your application. See [Software Component Lifecycle Management](#) [page 1068].

**Parent topic:** Getting Started in the ABAP Environment [page 35]

## Related Information

[Getting Started with a Trial Account in the ABAP Environment](#) [page 35]

[Learning Journey](#)

[Trial Learning Journey](#)

[Product Page](#)

[SAP Community](#)

[Solution Overview and Roadmap](#)

[SAP Road Map Explorer](#)

[Starter Scenario: Develop an SAP Fiori App Using ABAP](#)

[Tutorials](#)

[Video Tutorials](#)

## 2.3.2.1 Using a Booster to Automate the Setup of the ABAP Environment

You can use a booster to automate some of the required steps for setting up the ABAP environment in SAP Cloud Platform. 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 Cloud Platform. For more information, see [Boosters](#) in the SAP Cloud Platform documentation on SAP Help Portal.

### Procedure

1. Log on to the SAP Cloud Platform cockpit and choose the global account for the Cloud Foundry environment as administrator.
2. In the menu of the SAP Cloud Platform cockpit, choose [Booster](#).
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 Booster](#).
5. Follow the instructions in the steps of 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 organization for the ABAP environment are available.
- A service instance for the ABAP environment has been created.
- The entitlement [16 GB ABAP Runtime, 64 GB Database](#) has been configured.
- Quotas for the ABAP runtime, the required destination service, and the application runtime have been increased by 1.
- You're subscribed to the Web access for ABAP SaaS application and get direct browser access to your instances in the ABAP environment.
- An instance for the destination service is created, which applications in the ABAP environment need for outbound connectivity.
- Additional users as Cloud Foundry organization administrator and space manager have been added.

### i Note

These tasks are still part of this documentation, but there's no need to perform them if you have run the booster successfully. A note at the beginning of each task indicates that you can skip steps if you've used the booster.

## 2.3.2.2 Create an ABAP System

Learn how to create an ABAP system using the cockpit.

### Prerequisites

You need to increase the quota for the ABAP runtime. See [Increasing the Quota for the ABAP Runtime](#).

### Procedure

1. Navigate to the space in which you want to create your ABAP system. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).
2. In the navigation area, select ► **Services** ► **Service Marketplace**.   
You see a list of all services that are available to you.
3. Select **ABAP System**.
4. In the navigation area, choose **Instances**.  
You see a list of all instances that have already been created.
5. Select **New Instance**.  
A wizard opens that helps you create your instance.
6. In the **Choose Service Plan** section, select the **standard** plan and choose **Next**.
7. In the **Specify Parameters (Optional)** section, enter the email address of the administrator in the JSON format, as illustrated below, or browse a JSON file from your computer, and select **Next**.

```
{  
    "admin_email": "administrator@example.com",  
    "description": "Test System for Unit Tests",  
    "is_development_allowed": true,  
    "size_of_runtime": "1",  
    "size_of_persistency": "4",  
    "sapSystemname": "H01"  
}
```

### i Note

The email address is used to create the initial user for the system automatically including the assignment of the administrator role. The system can only be accessed with the specified user.

The description and system name are optional and user-defined. You can specify a description and a three-character name of your choice for your system to make it easier to refer to the corresponding system in the development environment. Or simply use the default name H01. The system name does not have to be technically unique.

With parameter `is_development_allowed`, you can control the changeability of development objects in the system. Please use the default setting `true` for your development system. If you want to protect all customer-related software components and ABAP namespaces against manual changes via ABAP Development Tools, use `is_development_allowed = false`. This setting is typically used for test and productive systems, where changes shall be imported only.

Parameter `size_of_runtime` is part of the quota plan `abap/abap_compute_unit`, with the ABAP compute unit representing the number of 16 GB blocks. The following sizes are available: 1, 2, 4, 6, 8 (16, 32, 64, 96, 128 GB RAM).

Parameter `size_of_persistency` is part of the quota plan `abap/hana_compute_unit`, with the HANA compute unit representing the number of 16 GB blocks. The following sizes are available: 4, 8, 16 (64, 128, 256 GB RAM).

8. Skip the [Assign Application \(Optional\)](#) section.
9. In the [Confirm](#) section, enter an instance name in the mandatory field, and confirm your settings by selecting [Finish](#).

#### i Note

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, without taking lower or upper case and special characters into account.

The ABAP system is being set up, which takes several minutes. The status is shown in the [Last Operation](#) column as [Creating](#). It takes another several minutes for the system to be configured automatically. After the setup is completed, an email is sent to the administrator specified in step 7.

## Related Information

[Development in the ABAP Environment \[page 385\]](#)

[Video Tutorial: Provision Instance](#) 

### 2.3.2.3 Define a Developer Role

Learn how to define a developer business role and set write access.

## Prerequisites

You are an administrator in the account.

## Procedure

1. In the SAP Fiori Launchpad, select the *Maintain Business Roles* tile in the *Identity and Access Management* section.
2. To define a new developer business role, choose *Create from Template*.
3. In the *Create Business Role from Template* dialog, use the value help to select template *SAP\_BR\_Developer* and choose *OK*.

**i Note**

The entries for *New Business Role ID* and *New Business Role* are automatically filled in.

4. Select *Maintain Restrictions*.
5. In the *Restrictions* tab, navigate to *Write* and select *Unrestricted* from the dropdown list.
6. Select *Back to Main Page* and choose *Save* and *Activate*.

**i Note**

Once the developer role has been activated, the Lifecycle Status is set to *Active*.

## Results

You have created a developer business role with unrestricted write access.

## Related Information

[Maintain Business Roles \[page 1043\]](#)

[How to Create a Business Role from a Template \[page 1045\]](#)

[Assigning the ABAP Developer User to the ABAP Developer Role](#)

## 2.4 Getting Started in the Kyma Environment

The getting started document describes the full list of steps you must complete as an administrator 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 [Provision Kyma Environment \[page 1083\]](#).

## Administrator Operations

As an administrator, perform the following steps to have a fully operational Kyma environment for the developers to work with. When steps 1-3 are already performed, the developers can start working on their Functions.

No.	Step	Description
1.	<a href="#">Provision Kyma Environment [page 1083]</a>	Set up a Kubernetes cluster with the project "Kyma" to connect and extend SAP systems.
2.	<a href="#">Roles in the Kyma Environment [page 1080]</a>	As a KymaRuntimeNamespaceAdministrator, you can assign the role of a KymaRuntimeNamespaceDeveloper. This way, you allow the administrators to manage Kyma and the developers to create Functions.
		<p><b>i Note</b></p> <p>Assign the roles before the users start using the Kyma Console. Not granting the roles results in an error.</p>
3.	<a href="#">Register an SAP Customer Experience System [page 707]</a> <a href="#">Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment [page 654]</a>	You can now start using the Kyma environment to integrate external systems. Your options are as follows: <ul style="list-style-type: none"><li>Integrate a CX system and the Kyma environment so that the Functions you develop can use the system's API and receive business events.</li><li>Integrate an SAP S/4 HANA Cloud system to use the services it provides to extend your applications.</li></ul>
4.	<a href="#">Assigning SAP Systems to a Formation [page 652]</a>	Group several solutions into one formation to meet the requirements of your business scenario.

## Developer Operations

Once the administrator sets up the environment, the developers can access the Kyma environment through the Console URL link that is available on the given subaccount. Upon logging, developers can start creating extensions for the SAP systems either from the Console UI or from the terminal after downloading the kubeconfig file with the cluster configuration. For details, see [Development in the Kyma Environment \[page 637\]](#).

# 3 Development

Develop and run business applications on SAP Cloud Platform. Use our application programming model, APIs, services, tools and capabilities.

We recommend using the application programming model for SAP Cloud Platform for full-stack application development. This model simplifies your development process by enabling you to create concise and comprehensive data and service models based on Core Data and Services (CDS). This modular approach means that you can reuse models and extend services to add features specific to your business logic. Thanks to the generic service provider, boilerplate code is reduced and code is simplified. This allows you to focus on your business logic and makes reviewing and maintaining your code easier.

Also, using our application programming model means you get languages, libraries and APIs that guide and support you through your development project. You can choose to develop stand-alone business applications or extend other cloud solutions, like SAP S/4 HANA or SAP SuccessFactors.

For more information, see [Developing with the SAP Cloud Application Programming Model \[page 54\]](#).

With SAP Cloud Platform, you are also free to choose your own approach. We support various programming languages on the Cloud Foundry and ABAP environment. You can also develop Multitarget Applications (MTA) on these environments. We provide information about developing, configuring and deploying your applications depending on your preferred programming language and development approach.

For more information, choose your environment and preferred programming language:

Supported Environments and Programming Languages

Environment	Programming Language
Cloud Foundry environment	<a href="#">SAP HANA [page 64]</a>
	<a href="#">Java [page 146]</a>
	<a href="#">Node.js [page 224]</a>
	<a href="#">Python [page 237]</a>
	<a href="#">SAPUI5 [page 248]</a>
	<a href="#">HTML5 [page 52]</a>
ABAP environment	<a href="#">ABAP [page 385]</a>

## [Development in the Cloud Foundry Environment \[page 53\]](#)

Learn more about developing applications on the SAP Cloud Platform Cloud Foundry environment.

## [Development in the ABAP Environment \[page 385\]](#)

Learn more about developing applications in the ABAP environment.

## [Development in the Kyma Environment \[page 637\]](#)

The Kyma environment allows you to extend existing SAP systems with your own Functions or microservices.

## Related Information

[SAP Cloud Platform Planning and Lifecycle-Management Guide](#)  
[Consuming Services in SAP Cloud Platform](#)

## 3.1 Development in the Cloud Foundry Environment

Learn more about developing applications on the SAP Cloud Platform Cloud Foundry environment.

### Overview

SAP Cloud Platform 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, Python, and Node.js 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 cockpit and dev-ops capabilities.
<b>Optimize development and operations</b>	Use the rich set of SAP Cloud Platform Cloud Foundry services including messaging, persistence, and many other capabilities.
<b>Use the application programming model</b>	Use languages, libraries and APIs tailored for full-stack development on SAP Cloud Platform.

If you have already monolithic applications running on SAP Cloud Platform and are looking for a way to run them on the Cloud Foundry environment, read our best practice guide. See [Migrating from the Neo Environment to the Multi-Cloud Foundation](#).

### Related Information

[Cloud Foundry Environment](#)

### 3.1.1 Developing Your First Application on SAP Cloud Platform

Follow any of these tutorials to develop your first application on SAP Cloud Platform. Choose your preferred programming model and technology.

Name	Technology
<a href="#">Use the Application Programming Model in our Starter Scenario</a>	Application Programming Model (CDS and Node.js)
<a href="#">Develop a Simple Hello World Application Using SAPUI5</a>	SAPUI5
<a href="#">Create a Node.js Application [page 226]</a>	Node.js
<a href="#">Authentication Checks in Node.js Applications [page 228]</a>	Node.js
<a href="#">Authorization Checks in Node.js Applications [page 232]</a>	Node.js
<a href="#">Create a Python Application [page 238]</a>	Python
<a href="#">Consume Cloud Foundry Services [page 241]</a>	Python
<a href="#">Authentication Checks in Python Applications [page 243]</a>	Python
<a href="#">Authorization Checks in Python Applications [page 246]</a>	Python

### 3.1.2 Developing with the SAP Cloud Application Programming Model

The SAP Cloud Application Programming Model 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 APIs tailored for full-stack development on SAP Cloud Platform.

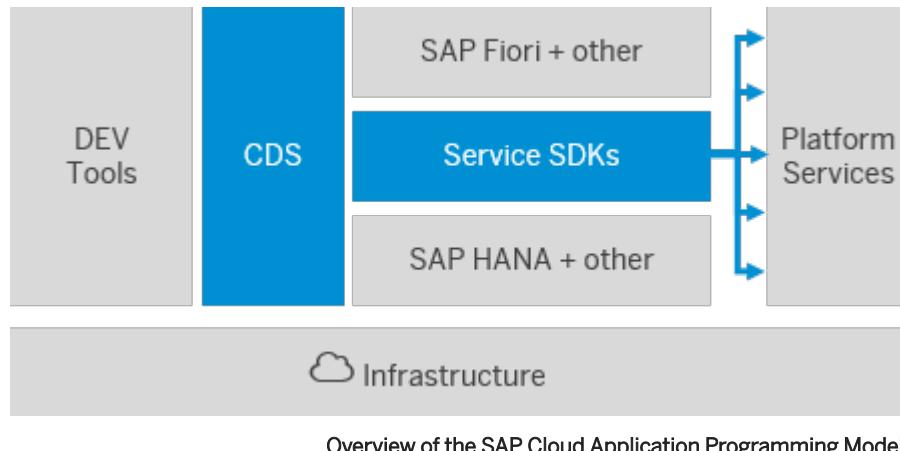
Use Core Data & Services (CDS) to build data models and service definitions on a conceptual level. These CDS models are used as inputs for the data, service, and UI layers. They're then translated to native artifacts, for example SQL database schemas, and interpreted to automatically serve requests at 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 Cloud 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 SAP Business Application Studio.

The following graphic shows the relationship between the application programming model, SAP Cloud Platform, platform services, and development tools:



#### [Best Practices \[page 55\]](#)

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 \[page 56\]](#)

CDS is the backbone of the SAP Cloud Application Programming Model. 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 \[page 57\]](#)

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 \[page 57\]](#)

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 \[page 58\]](#)

Find additional references in the following sections.

### 3.1.2.1 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](#) - This guide is about consuming services in general, including Local Services, External Services, and Databases.

- [Using Databases](#) - This guide is about defining, providing, implementing, deploying, and publishing services — so about Service Providers in general.
- [Serving Fiori UIs](#) - This guide explains how to add one or more SAP Fiori elements apps to a CAP project, how to add SAP Fiori elements annotations to respective service definitions, etc.
- [Authorization](#) - About restricting access to data by adding respective declarations to CDS models, which are then enforced in service implementations.
- [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.
- [Deploying to Cloud](#) - How to deploy your services to the cloud, for example, to SAP Cloud Platform.
- [Extensibility](#) - How SaaS subscribers can extend SaaS applications on a tenant level.
- [Multitenancy](#) - How to implement multitenant aware applications (SaaS applications) with CAP.
- [Native SAP HANA](#) - How to use existing database objects with CDS.

### 3.1.2.2 Core Data and Services (CDS) Language Reference Documentation

CDS is the backbone of the SAP Cloud Application Programming Model. 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:

- [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.
- [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.
- [OData Annotations](#) - Overview of OData Vocabularies by OASIS and SAP as well as instructions how to add OData annotations to CDS models and how they're mapped to EDMX outputs.

- [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.

### 3.1.2.3 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.

- [Overview](#)
- [Using Local Development](#)
- [Starting a New Project](#)
- [Working in Eclipse](#)
- [Stack Architecture](#)
- [Providing Service](#)
- [Consuming Services](#)
- [Model Reflection API](#)
- [Query Builder API](#)
- [Query Introspection API](#)
- [Working with Data](#)
- [Multitenancy](#)
- [Advanced Concepts](#)
- [Development](#)
- [Migration](#)

### 3.1.2.4 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](#)
- [Starting a New Project](#)
- [Defining Domain Models](#)
- [Defining Services](#)

- [Using Databases](#)
- [Adding/Serving UIs](#)
- [Adding Custom Logic](#)
- [Testing with Postman](#)
- [Deploying to Cloud](#)
- [Node.js Core Services API](#)
- [Authentication](#)
- [Best Practices](#)

### 3.1.2.5 References

Find additional references in the following sections.

#### 3.1.2.5.1 Sample Projects

Get started with samples and reusable packages created based on SAP Cloud Application Programming Model.

The SAP Cloud Application Programming Model 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 APIs tailored for full-stack development on SAP Cloud Platform. The samples for [Node.js](#) and [Java](#) provided can be run in a local setup on SQLite database.

#### 3.1.2.5.2 Tutorials

Learn how to develop business applications using the SAP Cloud Application Programming Model.

- Learn how to create a business service using CAP for Java and deploy it to SAP Cloud Platform: [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 Cloud Platform: [Create a Business Service with Node.js using Visual Studio Code](#).
- Learn how to use the application programming model and SAP Cloud SDK to extend SAP S/4HANA: [Use CAP and SAP Cloud SDK to Extend S/4HANA](#).
- Find more tutorials using the SAP Cloud Application Programming Model in the [Tutorial Navigator](#).

#### 3.1.2.5.3 Troubleshooting

If you run into issues working with the SAP Cloud Application Programming Model samples or tutorials, please refer to the Troubleshooting guide.

In the [Troubleshooting](#) guide, you can find a collection of frequently asked questions and provided solutions.

Furthermore you can get support as mentioned in section [Resources](#).

### 3.1.2.5.4 Community, Blogs & Q&A

Learn more about the SAP Cloud Application Programming Model 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.

### 3.1.2.5.5 What's New

This section provides information about what is new and what has changed in SAP Cloud Application Programming Model since the last release.

- [Release Notes](#)
- [Changelog](#)

## 3.1.3 APIs in the Cloud Foundry Environment

Access SAP Cloud Platform APIs through the SAP API Business Hub.

The central repository for APIs from SAP and selected partners is the SAP API Business Hub. It contains reference documentation for a variety of REST and ODATA API packages that enable you to leverage the capabilities of SAP Cloud Platform.

- For more information on the core services needed for development in the Cloud Foundry environment, check out the SAP Cloud Platform API package on the [SAP API Business Hub](#).
- For an overview of all API offerings for SAP Cloud Platform, see [SAP Cloud Platform on SAP API Business Hub](#).

If you're looking for information on APIs for a specific service, you can also use the [SAP Cloud Platform Regions and Services Portfolio](#). You can check the availability of your desired service in your region, then click on the service name to access its product page. If the service has APIs, you can find links to their documentation on the product page.

## Related Information

[SAP API Business Hub in the SAP Community](#)

### 3.1.4 Business Application Pattern

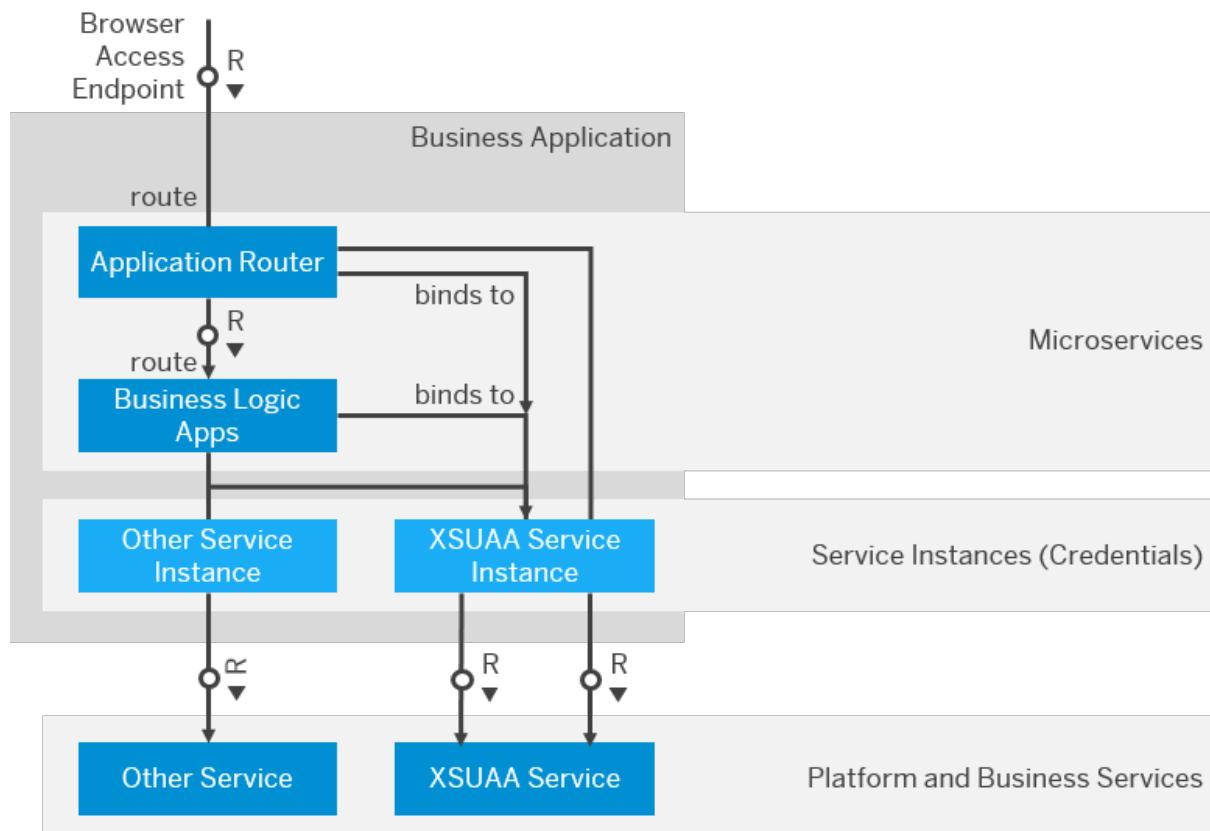
In the Cloud Foundry environment, SAP is promoting a pattern for building applications. We use the term **Business Application** to distinguish from single applications in the context of the Cloud Foundry environment.

This topic covers the following:

- [Application Patterns \[page 60\]](#)
- [Application Router \[page 61\]](#)
- [UAA Service \[page 61\]](#)
- [Platform and Business Services \[page 62\]](#)
- [Application Deployment \[page 62\]](#)

### 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 Cloud Platform 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 Cloud Platform 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 \[page 286\]](#) (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).

### 3.1.5 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 \[page 286\]](#). See [Multitarget Application Commands for the Cloud Foundry Environment \[page 938\]](#) for information about MTA deployment.

## Related Information

[Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#)

[Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#)

[Multitarget Application Commands for the Cloud Foundry Environment \[page 938\]](#)

## 3.1.6 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 auth-method 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 SAP Cloud Platform Cloud Foundry environment, see [Cloud Foundry Environment](#)
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](#)).

### 3.1.7 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 Cloud Platform 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 Cloud Platform 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.  See <a href="#">SAP HANA Cloud [page 65]</a>
You're using SAP Cloud Platform, SAP HANA service 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.  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 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 can find another collection of these features and elements in the following SAP Note: <a href="#">2868742</a>
	Have a look at the section <a href="#">HDI features are not available with SAP HANA Cloud</a> , which is the most important part in this scenario.
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</a>

## SAP HANA Cloud

This section is a quick introduction to the latest SAP HANA offering on SAP Cloud Platform 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 Web IDE Full-Stack and SAP HANA Cockpit.

- SAP Web IDE Full-Stack  
SAP Web IDE Full-Stack is the recommended tool to develop your SAP HANA database artifacts, see [SAP Web IDE Full-Stack > Getting Started](#)  
For an SAP Cloud Application Programming Model project, we recommend starting your project in SAP Web IDE Full-Stack and synchronize it with GitHub, see [SAP Web IDE Full-Stack > Using Source Control \(Git\)](#)  
That way, you leverage the power of SAP Web IDE for developing database artifacts. At the same time you can use the more mature tooling for service development, which resides in Visual Studio Code and SAP Business Application Studio.
- SAP HANA Cockpit  
The SAP HANA Cockpit is used to monitor and administer your SAP HANA instances, see [Getting Started with SAP HANA Cockpit](#)  
To learn how to access the SAP HANA Cockpit from the SAP Cloud 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](#)

## 3.1.8 Developing HTML5 Applications in the Cloud Foundry Environment

SAP Cloud Platform 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.

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 in SAP Cloud Platform Portal, build the URL, and define custom domains.

On the Cloud Foundry environment of SAP Cloud Platform you can run an application that was uploaded to HTML5 application repository using one of the following options: standalone application router or HTML5 application runtime managed by SAP Cloud Platform. 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 option that is managed by SAP Cloud Platform 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

Therefore, we recommend using the standalone Application Router only in advanced cases, for example when application router extensibility is required.

For more information, see [Developing HTML5 Applications Managed by SAP Cloud Platform](#).

## Related Information

[HTML5 Application Repository \[page 67\]](#)

[Application Router \[page 77\]](#)

### 3.1.8.1 HTML5 Application Repository

HTML5 application repository enables central storage of HTML5 applications' static content on the SAP Cloud Platform Cloud Foundry 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 in the SAP Cloud Platform Cloud Foundry environment.

#### 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.

## Restrictions

- 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.

## See Also

If you want to learn more about services in the SAP Cloud Platform Cloud Foundry environment, see [Using Services in the Cloud Foundry Environment \[page 370\]](#).

### 3.1.8.1.1 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.

## Related Information

[Deploy Content Using Generic Application Content Deployer \[page 69\]](#)

[Deploy Content Using HTML5 Application Deployer \[page 70\]](#)

[Redeploy Content \[page 73\]](#)

[Undeploy Content \[page 74\]](#)

### 3.1.8.1.1.1 Deploy Content Using 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 \[page 931\]](#).
- 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 Plug-in in the Cloud Foundry Environment \[page 937\]](#).

#### 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 \*.mtar file with following structure:

```
↳ Sample Code  
myMtar.mtar  
  ...
```

```
mydeployer  
  content.zip  
META-INF  
  mtad.yaml  
MANIFEST.MF
```

3. In the MANIFEST.MF file, define the following section for the deployer:

#### ↳ Sample Code

```
Manifest-Version: 1.0  
Created-By: SAP WebIDE  
Name: mydeployer/content.zip  
MTA-Module: ui_deployer  
Content-Type: application/zip
```

4. In the mtad.yaml file, define the deployer module:

#### ↳ Sample Code

```
ID: testdeployer  
_schema-version: '3.1'  
modules:  
- name: ui_deployer  
  type: com.sap.application.content  
  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
```

5. Deploy the \*.mtar using the CLI command cf deploy.

### 3.1.8.1.1.2 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 \[page 931\]](#).
- 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 Plug-in in the Cloud Foundry Environment \[page 937\]](#).

## Context

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  
    - app2  
      ...  
  package.json  
  manifest.yaml
```

- 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`.

## i Note

The format of the application version is `xx.xx.xx` to be compliant with the version element format of the MTA Descriptor. For more information, see [MTA Descriptor Elements](#)

## ↳ 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 `mtad.yaml` file:

- Under `modules`, add your app.
- Add a dependency to the `html5-apps-repo` service and the `app-host` service instance.

## ↳ 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

```

The `mtad.yaml` file acts as the deployment descriptor. For a general description of MTA descriptors, see [Multitarget Applications in the Cloud Foundry Environment \[page 286\]](#).

4. Create the `pom.xml` file for your project according to the multi-target application build (MBT) requirements.
5. In the Cloud Foundry command line interface (CLI), navigate to your project root and enter the CLI command: `mvn clean install`.  
The `*.mtar` file is generated.
6. Deploy the `*.mtar` file using the CLI command `cf deploy`.

```
cf deploy myHTML5App-deployer-assembly-0.0.1.mtar
```

The Cloud Foundry deploy plug-in uses the `mtad.yaml` file to configure 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.

### 3.1.8.1.1.3 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 these versions 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
```

### 3.1.8.1.1.4 Undeploy Content

To undeploy content you need to delete the content from the repository and delete the `app-host` service plan instance.

#### Procedure

- Open the Cloud Foundry command line interface (CLI).
- Undeploy the `*.mtar` file using the CLI command: `cf undeploy` and add the `--delete-services` option.

Example: `cf undeploy html5.repo.deployer.myHTML5App --delete-services`

- 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.

#### i 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: `cd delete-service SERVICE-NAME`.

### 3.1.8.1.2 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 \[page 68\]](#).
- *app-runtime*  
Use this service plan to consume HTML5 applications stored in the repository. For more information, see [Consuming Content \[page 75\]](#).

### 3.1.8.1.3 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": "MyHtml5AppsRepoRuntimeInstance",  
  "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>`

#### i 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.

### 3.1.8.1.4 Security and Data Privacy Configuration

This section provides information about security and data privacy.

- Identity Provider and Identity Management

The User Account and Authentication (UAA) server is responsible 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.

- 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.

## 3.1.8.2 Application Router

The application router is the single point-of-entry for an application running in the Cloud Foundry environment on SAP Cloud Platform. 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 application router is a library that is available on [npmjs.com \(NPM\)](#). For more information, see [@sap/approuter](#).

### Related Information

[Setting Up the Application Router \[page 77\]](#)

[Resource Files \[page 80\]](#)

[Application Router Configuration \[page 81\]](#)

[Routing Configuration File \[page 86\]](#)

[Application Routes and Destinations \[page 111\]](#)

[Environment Variables \[page 121\]](#)

[Multitenancy \[page 131\]](#)

[Extending the Application Router \[page 133\]](#)

[Extension API of the Application Router \[page 136\]](#)

### 3.1.8.2.1 Setting Up the Application Router

This section describes how to set up the application router.

#### Prerequisites

- cf CLI is installed locally, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- 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://npm.sap.com`

#### Context

The application router is configured in the `xs-app.json` file. The `package.json` file contains the start command for the application router and a list of package dependencies.

## 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/                                # Application descriptors
|   | - xs-app.json                      # Application routes configuration
|   | - package.json                     # Application router details/dependencies
|   \- resources/
```

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 \[page 86\]](#).

- 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/.*",
      "target": "index.html"
    }
  ]
}
```

```

        "destination": "backend"
    },
{
    "source": "^/(.*)",
    "localDir": "resources"
}
]
}

```

- b. Add the routes (destinations) for the specific application (for example, node-hello-world) to the env section 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
  host: myHost-node-hello-world
  domain: 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
      }
    ]

```

6. 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.

### 3.1.8.2.2 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
<code>package.json</code>	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
<code>xs-app.json</code>	The application descriptor contains the configuration used by the application router (for example, destinations for request forwarding)	Yes
<code>resources/</code>	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
<code>local-destinations.json</code>	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.	No
<code>default-services.json</code>	Provides the required destinations information for local development	-
	Defines the configuration for one or more special User Account and Authentication (UAA) services for local development	-

→ Tip

If you have a static resources folder name in the `xs-app.json` file, we recommend that you use `localDir` as default.

#### ↳ Sample Code

```
xs-app.json

{
  "source": "^/web-pages/(.*)$",
  "localDir": "my-static-resources"
}
```

### 3.1.8.2.3 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”.

The application router configuration is

#### → 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

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.

#### i 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.

#### i 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 \[page 86\]](#).

## 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.

### i 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.

## 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 \[page 85\]](#) 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 is 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"
}
```

### i 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 Cloud Platform connectivity service. The connectivity service enables you to manage proxy access to SAP Cloud Platform 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 `@sap/logging` package, which means that all of the typical logging features are available to control application logging. For example, to set all logging and tracing to the most detailed level, set the `<XS_APP_LOG_LEVEL>` environment variable to “debug”.

### i Note

Enabling 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.

### → 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”.

You can enable additional traces of the incoming and outgoing requests by setting the environment variable `<REQUEST_TRACE>` to true. When enabled basic information will be logged for every incoming and outgoing request of the application router.

The `@sap/logging` package sets the header `x-request-id` in the application router's responses. This is useful if you want to search the application router's logs and traces for entries that belong to a particular request execution. Note that the application router does not change the headers received from the back end and forwarded to the client. If the back end is a `Node.js` application which uses the `@sap/logging` package (and also sets the `x-request-id` header), then the value of the header that the client receives is the one coming from the back end and not the one from the application router itself.

## Related Information

[Routing Configuration File \[page 86\]](#)

[Application Router \[page 77\]](#)

[Resource Files \[page 80\]](#)

### 3.1.8.2.4 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": "employeeServices",  
      "csrfProtection": false  
    }  
  ]  
}
```

```

    "authenticationType": "xsuaa",
    "scope": ["$XSAPPNAME.viewer", "$XSAPPNAME.writer"],
    "csrfProtection": true
  },
  {
    "source": "^/(.*)$",
    "target": "/web/$1",
    "localDir": "static-content",
    "replace": {
      "pathSuffixes": ["/abc/index.html"],
      "vars": ["NAME"]
    }
  }
],
"login": {
  "callbackEndpoint": "/custom/login/callback"
},
"logout": {
  "logoutEndpoint": "/my/logout",
  "logoutPage": "/logout-page.html"
},
"destinations": {
  "employeeServices": {
    "logoutPath": "/services/employeeService/logout",
    "logoutMethod": "GET"
  }
},
"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 89]	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 89]	String	The method used to authenticate user requests, for example: "route" or "none" (no authentication).
<a href="#">sessionTimeout</a> [page 90]	Number	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.
<a href="#">routes</a> [page 90]	Array	Defines all route objects, for example: source, target, and, destination.

Property	Type	Description
<a href="#">login [page 102]</a>	Object	A redirect to the application router at a specific endpoint takes place during OAuth2 authentication with the User Account and Authentication service (UAA).
<a href="#">logout [page 102]</a>	Object	You can define any options that apply if you want your application to have central log out end point.
<a href="#">destinations [page 104]</a>	Object	Specify any additional options for your destinations.
<a href="#">services [page 104]</a>	Object	Specify options for a service in your application.
<a href="#">compression [page 105]</a>	Object	The compression keyword enables you to define if the application router compresses text resources before sending them.
<a href="#">pluginMetadataEndpoint [page 106]</a>	String	Adds an endpoint that serves a JSON string representing all configured plugins.
<a href="#">whitelistService [page 107]</a>	Object	Enable the white-list service to help prevent against click-jacking attacks.
<a href="#">websockets [page 108]</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 109]</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 \[page 77\]](#)

[Application Router Configuration \[page 81\]](#)

### 3.1.8.2.4.1 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
welcomeFile	String	No	HTML page, for example, index.html

#### ↳ Code Syntax

```
"welcomeFile": "index.html"
```

### 3.1.8.2.4.2 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.

### 3.1.8.2.4.3 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.

#### i Note

The `sessionTimeout` property is no longer available; to set the session time out value, use the environment variable `<SESSION_TIMEOUT>`.

#### ↳ 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.

### 3.1.8.2.4.4 routes

Defines all route objects, for example: source, target, and, destination.

- [Routes Properties \[page 90\]](#)
- [Route Configuration Examples \[page 96\]](#)

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>

#### i Note

Be aware that RegEx is applied to the full URL, including query parameters.

Property	Type	Mandatory	Description
<a href="#">httpMethods [page 93]</a>	Array of upper-case HTTP methods	No	HTTP methods that are served by this route; the supported methods are: DELETE, GET, HEAD, OPTIONS, POST, PUT, TRACE, and PATCH.
			<p><b>→ Tip</b></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.
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 111]</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 94]</a>	String	No	The directory from which application router serves static content (for example, from the application's web/ module).
<a href="#">replace [page 94]</a>	Object	No	An object that contains the configuration for replacing placeholders with values from the environment.
			<p><b>i Note</b></p> <p>It is only relevant for static content.</p>
authenticationType	String	No	The value can be "xsuaa", "basic" or "none". The default value is "xsuaa". When "xsuaa" is used, the specified UAA server handles the authentication (the user is redirected to the UAA's logon form). If "none" is used then no authentication is needed for this route.

Property	Type	Mandatory	Description
csrfProtection	Boolean	No	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.
scope	Array/String/ Object	No	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> .
cacheControl	String	No	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.  The cacheControl property is only effective when one of the following settings is performed: <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>
identityProvider	String	No	The name of the identity provider you use if it's provided in route's definition. If it isn't provided, the route is authenticated with the default identity provider.
<p><b>i Note</b></p> <p>If <code>authenticationType</code> is set to <code>Basic Authentication</code> or <code>None</code>, don't define the <code>identityProvider</code> property.</p>			

## i Note

Route order is important. The first matching route will be used. Therefore, it is recommended to sort the routes by 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"
    }
  ]
}
```

#### i 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
pathSuffixes	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.
vars	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.

### i 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>`.

## i 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"  
}
```

### i 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`.

### i 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.

#### i 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 the url should be defined in the following way:

```
https://<tenant>-<destination>.<customdomain>/<pathofile>
```

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/([^/]+)/",  
  "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.

### i 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".

### i 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, and OPTIONS. 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.

### i 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`.

### i 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.

### 3.1.8.2.4.5 login

A redirect to the application router at a specific endpoint takes place during OAuth2 authentication with the User Account and Authentication service (UAA).

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”.

### 3.1.8.2.4.6 logout

You can define any options that apply if you want your application to have central log out end point.

In this object you can define an application's central log out end point by using the `logoutEndpoint` property, as illustrated in the following example:

#### « Sample Code

```
"logout": {  
    "logoutEndpoint": "/my/logout"  
}
```

Making a GET request to “/my/logout” triggers a client-initiated central log out. Central log out can be initiated by a client or triggered due to a session timeout, with the following consequences:

- Client-initiated central log out
  - 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).
  - Redirects to the User Account and Authentication service (UAA), if such a service is provided, and logs out from there.
- Session time out
  - 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).

You can use the `logoutPage` property to specify the Web page in one of the following ways:

- URL path

The UAA service redirects the user back to the application router, and the path is interpreted according to the configured routes.

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
```

#### i 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 UAA will redirect the user to a page (or application) different from the application router.

#### ↳ Sample Code

```
"logout": {
```

```
        "logoutEndpoint": "/my/logout",
        "logoutPage": "http://acme.com/employees.portal"
    }
```

### 3.1.8.2.4.7 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.
<code>logoutMethod</code>	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

### 3.1.8.2.4.8 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": "countryservice",
        "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 VCAP_SERVICES that contains the URL of the service.
logoutPath	String	No	The log out end point for your destination. The logoutPath will be called when central log out is triggered or a session is deleted due to a time out. The request to logoutPath contains additional headers, including the JWT token.
logoutMethod	String	No	The logoutMethod property specifies the HTTP method with which the logoutPath will be requested, for example, POST, PUT, GET; the default value is POST

## Related Information

[Integration with Business Services \[page 116\]](#)

### 3.1.8.2.4.9 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
minSize	Number	No	Text resources larger than this size will be compressed.
enabled	Boolean	No	Globally disables or enables compression. The default value is true.

**i Note**

If the <COMPRESSION> environment variable is set it will overwrite any existing values.

### 3.1.8.2.4.10 pluginMetadataEndpoint

Adds an endpoint that serves a JSON string representing all configured plugins.

**↳ Sample Code**

```
{  
    "pluginMetadataEndpoint": "/metadata"  
}
```

**i Note**

If you request the relative path /metadata of your application, a JSON string is returned with the configured plug-ins.

### 3.1.8.2.4.11 whitelistService

Enable the white-list service to help prevent against click-jacking attacks.

Enabling the white-list service 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 white-list service is enabled in the application router, each time an HTML page needs to be rendered in a frame, the white-list service is used check if the parent frame is allowed to render the content in a frame.

### Host Names and Domain Names

The white-list service reads a list of allowed host names and domains defined in the environment variable `<CJ_PROTECT_WHITELIST>`; the content is a JSON list of objects with the following properties:

White List 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.

The following snippet shows an example of the resulting JSON array:

#### ↳ Sample Code

```
[  
  {  
    "protocol": "http",  
    "host": "* acme.com",  
    "port": 12345  
  },  
  {  
    "host": "hostname.acme.com"  
  }  
]
```

#### i Note

Matching is done against the properties provided. For example, if only host name is provided, the white-list service returns “framing: true” for all, and matching will be for all schemata and protocols.

## Return Value

The white-list service accepts only GET requests, and returns a JSON object as the response. The white-list 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 <CJ\_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 white-list 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.

## 3.1.8.2.4.12 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.

### **3.1.8.2.4.13 `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"}
]}
```

#### **i Note**

The contents of the `errorPage` configuration section have no effect on errors that are not generated by the application router.

### 3.1.8.2.5 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.

##### i 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 (Beta feature)

`x-approuter-authorization` header contains the JWT token to support the Service to Application Router scenario. The application router can receive a consumer service XSUAA JWT token 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 the request.

### i Note

The XSUAA JWT token is generated with the same xsuaa service instance that is bound to the application router.

### ⚠ Caution

You should not use SAP Cloud Platform beta features in subaccounts that belong to productive enterprise accounts. Any use of beta functionality is at the customer's own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

### 3.1.8.2.6 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.

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": "backend",
          "url": "http://<hostname>:<node-port>",
          "forwardAuthToken": true
        }
      ]
  services:
    - node-uaa
```

### i 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. <div style="border-left: 2px solid #4f81bd; padding-left: 10px;"><p><b>i Note</b></p><ul style="list-style-type: none"><li>• When using basic authentication, User and Password are mandatory.</li><li>• When using principal propagation, the proxy type is on-premise.</li><li>• When using OAuth2SAMLBearerAssertion, the uaa.user scope in the xs-security.json file is required.</li></ul></div>
ProxyType	Supported types: <ul style="list-style-type: none"><li>• on-premise If set, binding to SAP Cloud Platform connectivity service is required.</li><li>• internet</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 true, the OAuth token is sent to the destination. The default value is false. 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. <div style="border-left: 2px solid #4f81bd; padding-left: 10px;"><p><b>i Note</b></p><p>If the ProxyType is set to on-premise, don't set the ForwardAuthToken property to true.</p><p>If the Authentication is other than NoAuthentication, don't set the ForwardAuthToken property to true.</p></div>

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 ms.
	<p><b>i 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.

- 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 Cloud Platform Connectivity service. The connectivity service handles proxy access to the SAP Cloud Platform 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.

## Related Information

[Application Router \[page 77\]](#)

[Resource Files \[page 80\]](#)

[Consuming the Destination Service](#)

[Consuming the Connectivity Service](#)

### 3.1.8.2.7 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:

- This feature is only supported in Cloud Foundry.
- 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.
- The `destination` property is not supported.
- External session management via extensibility is not supported.
- 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

### 3.1.8.2.7.1 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 HTML5 Application Repository, must have the following format:

#### ↳ Sample Code

```
https://<tenantId>.<appRouterHost>.<domain>/<bsPrefix>.<appName-appVersion>/
<resourcePath>
```

For example:

#### ↳ Sample Code

```
https://tenant1.myapproouter.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.
<b>i Note</b>		
Must not contain dots or special characters.		
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.

### 3.1.8.2.8 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
user_token	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.
client_credentials	The application router generates a client_credentials 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
sap.cloud.service	Yes	Service name as referenced from xs-app.json route and business service prefix, if provided by the Business Service UI.
sap.cloud.service.alias	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.
endpoints	No	One or more endpoints that can be used to access Business Service data.
html5-apps-repo	No	The html5-apps-repo.app_host_id contains one or more html5-apps-repo 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 xsappname in the SaaS Registry getDependencies callback.
grant_type	No	The grant type that should be used to trigger requests to the Business Service. Allowed values: user_token (default) or client_credentials.

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.

Property	Type	Optional	Default	Description
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 \[page 118\]](#)

[Accessing Business Service UI \[page 120\]](#)

[services \[page 104\]](#)

### 3.1.8.2.8.1 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": "countryservice"
  },
]
```

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 \[page 116\]](#)

[Accessing Business Service UI \[page 120\]](#)

### 3.1.8.2.8.2 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.comsapappbasiccountrylist/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 `comsapappbasiccountrylist` 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 \[page 116\]](#)

[Accessing Business Service Data \[page 118\]](#)

### 3.1.8.2.9 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.
<code>cookies</code> [page 124]	Provides cookies that the application router returns to the client in its responses.
<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 X-Frame-Options 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_TIMEOUT</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 <code>Headers</code> [page 110].

Variable	Description
CORS	Provides support for cross-origin requests, for example, by allowing the modification of the request header.

## 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 a whitelist 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
name	String	Yes	A unique identifier for the destination

Property	Type	Mandatory	Description
url	String	Yes	The Unique Resource Locator for the application (microservice)
proxyHost	String	No	The host of the proxy server used in case the request should go through a proxy to reach the destination.
			<p><b>→ Tip</b></p> <p>Mandatory if <code>proxyPort</code> is defined.</p>
proxyPort	String	No	The port of the proxy server used in case the request should go through a proxy to reach the destination.
			<p><b>→ Tip</b></p> <p>Mandatory if <code>proxyHost</code> is defined.</p>
forwardAuthToken	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 purposed by back-end services.
strictSSL	Boolean	No	Configures whether the application router should reject untrusted certificates. The default value is "true".
			<p><b>⚠ Caution</b></p> <p>For testing purposes only. Do <b>not</b> use this property in production environments!</p>
timeout	Number	No	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><b>→ Tip</b></p> <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>

Property	Type	Mandatory	Description
proxyType	String	No	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.

→ Tip

In Cloud environments, if you set the application's destination `proxyType` to `onPremise`, a binding to the SAP Cloud Platform connectivity service is required, and the `forwardAuthToken` property must not be set.

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 cookies

If configured, the application router will send additional cookies in its responses to the client. Additional cookies can be set in the `<COOKIES>` environment variable.

Example of configuration for cookies in the `manifest.yml`:

#### ↳ Sample Code

```
env:  
  COOKIES: >  
    { "SameSite": "None" }
```

In this example, the application router sets the `SameSite` cookie attribute to `None` for the `JSESSIONID` cookie in the responses to the client.

#### i Note

Currently, only the `SameSite` cookie is supported.

## 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
```

#### i 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 white-list 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 white-list service to validate if the parent frame is allowed to render the requested content in a frame. The check itself is provided by the white-list service

The following example shows how to add a host name to the click-jacking protection white list 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:

White List 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.

### i Note

Matching is done against the properties provided. For example, if only the host name is provided, the white-list 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.

### **i 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>
```

### **i 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 60,000
```

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 120,000 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
```

## 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.

### i 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" } ].
<b>i Note</b>	<p>Matching is case-sensitive. In addition, if no port or protocol is specified, the default is "*".</p>		
The default configuration is: [ { "host": "*" } ], which means that the server allows <b>any</b> origin to access the resource.			
allowedMethods	Array	No	A comma-separated list of HTTP methods that are allowed by the server, for example, "GET", "POST". If allowMethods is defined but no method is specified, the default "GET", "POST", "HEAD", "OPTIONS" (all) applies.
<b>→ Tip</b>	<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 maxAge 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$"
        }
      ]

```

## Related Information

[Application Router \[page 77\]](#)

[Routing Configuration File \[page 86\]](#)

## 3.1.8.2.10 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" : "<approouter-host>/<getDependenciesPath>/<tenantId>",
    "onSubscription" : "<approouter-host>/<onSubscriptionPath>/<tenantId>"
  },
  "displayName" : "<display_name>",
  "description" : "<description>",
  "category" : "<category>"
}
```

Specify the following parameters:

Parameters	Description
<code>xsappname</code>	The <code>xsappname</code> configured in the security descriptor file used to create the XSUAA instance (see <a href="#">Develop the Multitenant Application [page 196]</a> ).
<code>getDependenciesPath</code>	(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.
<code>onSubscriptionPath</code>	This parameter 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 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).

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 this parameter is left empty, it's assigned to the default category.

### 3.1.8.2.11 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('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('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('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

- o 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('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('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 \[page 136\]](#)

### 3.1.8.2.12 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>xsappContext</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.

- `path` - handle only requests starting with this path (string, optional)
- `handler` - a middleware function to invoke (function, mandatory)

Returns this for chaining.

---

## Related Information

[Middleware Connect Framework ↗](#)

[Extending the Application Router \[page 133\]](#)

### 3.1.8.3 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-XS-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).

**i Note**

The credentials(clientid/clientsecret) should be removed.

- Upload your Application Router logs

## Related Information

[Getting Support for SAP Cloud Platform \[page 1161\]](#)

[Troubleshooting \[page 139\]](#)

### 3.1.8.4 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".

Term	Description
Cause	<p>In order to support subscriptions out of the box, a route with format *.&lt;custom-domain&gt; should be created and mapped to the application router. The * host represents a wildcard that is replaced by the actual subscriber subdomain during runtime.</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 is &lt;subdomain&gt;-&lt;myapprouter&gt;.&lt;scp domain&gt;. It means that you have to create a route with the expected subdomain for each new subscriber.</p>
Solution	<p>In development setups, if you don't have a custom domain before subscription, then create a route with the expected subscriber subdomain.</p>

#### 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	<p>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.</p>

#### 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."

Term	Description
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.

## 403: Failed to create app-host service instance even though app-host plan is visible

Term	Description
Issue	When trying to create an app-host service instance, you receive "Error creating service "dt_TestMTA_ui_deployer" from offering "html5-apps-repo" and plan "app-host": Controller operation failed: 403 Forbidden: A service instance for the selected plan cannot be created in this organization. The plan is visible because another organization you belong to has access to it".
Cause	The app-host service was not entitled to the subaccount associated to this organization. However, it was entitled to another subaccount under the same global account.
Solution	Entitle the HTML5 application repository plan app-host service to the current subaccount.

## 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.
Solution	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:

```
cf html5-delete --content <app-host-id> [...]
```

## 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.

Term	Description
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 <code>"public" : true</code> in the app <code>manifest.json</code> 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	The application name provided in URL is not correct. 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.
Solution	Check the application name.

## 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 xs-app.json but calls do not get routed and instead return a 404 error.
Cause	The routes in the xs-app.json 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 be reached.
Solution	Move the route leading to the html5-apps-repo-rt to be the last entry in the xs-app.json file.

## 500: Failed to retrieve xs-app.json

Term	Description
Issue	Serving content from the application router 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 approuter 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 Cloud Platform 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:  <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 Cloud Platform cockpit  <a href="#">Global Account</a>    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	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:

```
{
  "routes": [
    {
      "source": "^/ui/index.html",
      "target": "index.html",
      "service": "html5-apps-repo-rt",
      "authenticationType": "xsuaa"
      "cacheControl": "no-cache, no-store, must-revalidate"
    }
  ]
}
```

## 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;.scp_domain. 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.

## 3.1.9 Developing Java in the Cloud Foundry Environment

Find here selected information for Java development on SAP Cloud Platform Cloud Foundry environment and references to more detailed sources.

The SAP Java buildpack is a SAP Cloud Platform Cloud Foundry buildpack for running JVM-based applications. The buildpack provides the following runtimes: [Tomcat \[page 149\]](#), [TomEE \[page 151\]](#), [TomEE 7 \[page 154\]](#), and [Java Main \[page 157\]](#).

### Usage

To use this buildpack specify its name when pushing an application to the Cloud Foundry.

```
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>
  buildpack: 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
...
...
```

## Versioning

The SAP Cloud Platform Cloud Foundry environment provides four versions of the SAP Java Buildpack as part of its system buildpacks:

- `sap_java_buildpack` - Always holds the latest available version of the SAP Java Buildpack. All new features and fixes are provided with this version.
- `sap_java_buildpack_<version_latest>` - Holds the latest available version of the SAP Java Buildpack; available for a limited timeframe (four to six weeks).
- `sap_java_buildpack_<version_previous>` - This version used to be latest in the previous update of the Cloud Foundry environment; available for a limited timeframe (four to six weeks).
- `sap_java_buildpack_<version_before_previous>` - This version used to be latest before two updates of the Cloud Foundry environment; available for a limited timeframe (four to six weeks).

### Important considerations about the usage of the different versions of SAP Java Buildpack

- If you always use `sap_java_buildpack` - This is the way to go in order to take advantage of any new features and fixes in the SAP Java buildpack. Thus it is guaranteed that the buildpack is always available. The drawback in this case is the limited time for any adoption which might be needed. In such a scenario applications can fall back to an older version temporarily to avoid any down time.
- If you pin the version of the buildpack - developers should be aware of the fact that this version will exist for a limited amount of time. This may lead to the situation where a restage is failing because the used version of the buildpack is not available anymore. To avoid this, it is recommended to follow the updates of the buildpack and test the application with the newest buildpack so that it could be adopted in time, in case an adoption is required, and to update the version regularly. In this scenario developers should never allow their application to run on an outdated buildpack version.

For clarity below is an example how the version update will take place:

Provided that the latest version of the SAP Java Buildpack is 1.2.3, the output of the `cf buildpacks` command would be:

buildpack	position	enabled	locked	filename
sap_java_buildpack	a	true	false	
sap_java_buildpack-1.2.3.zip				
sap_java_buildpack_1_2_3	b	true	false	
sap_java_buildpack-1.2.3.zip				
sap_java_buildpack_1_2_2	c	true	false	
sap_java_buildpack-1.2.2.zip				

```
sap_java_buildpack_1_2_1      d      true      false  
sap_java_buildpack-1.2.1.zip
```

When the SAP Java buildpack is updated on the Cloud Foundry environment from v.1.2.3 to v.1.2.4 the list will change to:

```
buildpack      position  enabled  locked  filename  
sap_java_buildpack      a      true      false  
sap_java_buildpack-1.2.4.zip  
sap_java_buildpack_1_2_4    b      true      false  
sap_java_buildpack-1.2.4.zip  
sap_java_buildpack_1_2_3    c      true      false  
sap_java_buildpack-1.2.3.zip  
sap_java_buildpack_1_2_2    d      true      false  
sap_java_buildpack-1.2.2.zip
```

making `sap_java_buildpack_1_2_1` no longer available for applications.

### i Note

No fixes will be provided to the older versions of the buildpack. Fixes, including security fixes, will be part of the latest version.

## Switching to different version of Buildpack

To use `sap_java_buildpack_<version_suffix>` version of the buildpack, specify its name when pushing an application to Cloud Foundry:

```
cf push -f <PATH_TO_APP_MANIFEST> -b sap_java_buildpack_<version_suffix>
```

or use the `buildpack` attribute to specify it in the `manifest.yml`:

```
---  
applications:  
- name: application-name  
  memory: 128M  
  path: ./target/application-name.war  
  instances: 1  
  buildpack: sap_java_buildpack_<version_suffix>
```

or in the `mtad.yml` of your mtar archive:

```
...  
modules:  
- name: application-name  
  type: java.tomcat  
  path: ./target/application-name.war  
  properties:  
  ...  
parameters:  
  ...  
  memory: 512M  
  buildpack: sap_java_buildpack_<version_suffix>  
...
```

## Components

The SAP Java buildpack provides the following components (containers, jres, frameworks) in the application container (<APP\_ROOT\_DIR>/app/META-INF/.sap\_java\_buildpack):

- Runtime ([Tomcat \[page 149\]](#), [TomEE \[page 151\]](#), [TomEE 7 \[page 154\]](#), [Java Main \[page 157\]](#))
- [Memory Calculator V1 \(SAP JVM Memory Calculator\) \[page 165\]](#) - by default
- [Memory Calculator V2 \[page 164\]](#) - optional
- SAP JVM
- Log Level Client
- JVMKill Agent

### 3.1.9.1 Application Containers

The following application containers are available for usage with the SAP Java Buildpack.

- [Tomcat \[page 149\]](#)
- [TomEE \[page 151\]](#)
- [TomEE 7 \[page 154\]](#)
- [Java Main \[page 157\]](#)

#### 3.1.9.1.1 Tomcat

By default web applications pushed with the SAP Java buildpack are running in an Apache Tomcat container.

Applications could explicitly define the targeted application container by using the TARGET\_RUNTIME environment variable in the application manifest.yml file.

##### ↳ Sample Code

manifest.yml

```
---
applications:
- name: <APP_NAME>
  ...
  env:
    TARGET_RUNTIME: tomcat
```

## Provided APIs

The tomcat application runtime container provides the following standard APIs:

Runtime	Tomcat	Supported Specification Version
tomcat	Apache Tomcat 8	Java Servlets 3.1 Java ServerPages (JSP) 2.3 Expression Language (EL) 3.0 Debugging Support for Other Languages 1.0 Java API for WebSocket 1.1

## Customizing the SAP Java Buildpack Defaults

SAP Java Buildpack provides some default configurations for the Tomcat application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature.

Below is a list with all of the placeholders which could 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 which can be used to enable or disable 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

The SAP Java Buildpack provides the default configurations for unlimited sessions for the Tomcat application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature. To limit the number of active sessions set the *maxActiveSessions* attribute on a *Manager* element, for 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 *web.xml*:

```
<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 using *app\_context\_root* in the application *manifest.yml* file. For example:

```
...  
env:  
  JBP_CONFIG_TOMCAT: "[tomcat:{app_context_root: test_context_path}]"  
...
```

## 3.1.9.1.2 TomEE

By default web applications pushed with the SAP Java buildpack are running in an Apache Tomcat container.

Applications could explicitly define the targeted application container - Apache TomEE, by using the *TARGET\_RUNTIME* environment variable in the application *manifest.yml* file.

## ↳ Sample Code

manifest.yml

```
---
applications:
- name: <APP_NAME>
...
env:
  TARGET_RUNTIME: tomee
```

## Provided APIs

The [tomee](#) application runtime container provides the following standard APIs:

Runtime	TomEE	Supported Specification Version
tomee	Apache TomEE JAX-RS (Java EE 6) Java Servlets 3.0 Java ServerPages (JSP) 2.2 Expression Language (EL) 2.2 Debugging Support for Other Languages 1.0 Standard Tag Library for JavaServer Pages (JSTL) 1.2 Java API for WebSocket 1.1 Java ServerFaces (JSF) 2.0 Java Transaction API (JTA) 1.1 Java Persistence API (JPA) 2.0 Java Contexts and Dependency Injection (CDI) 1.0 Java Authentication and Authorization Service (JAAS) Java Authorization Contract for Containers (JACC) 1.3 JavaMail API 1.4 Bean Validation 1.0 Enterprise JavaBeans 3.1 Java API for RESTful Web Services (JAX-RS) 1.1	

## Customizing the SAP Java Buildpack Defaults

The SAP Java buildpack provides some default configurations for the Apache TomEE application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature.

Below is a list with all of the placeholders which could 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 which can be used to enable or disable the TRACE HTTP method	false

To configure the HTTP header size, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomee/conf/server.xml':  
  {'connector.maxHttpHeaderSize':1024}]"
```

To configure the maximum number of threads, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomee/conf/server.xml':  
  {'connector.maxThreads':800}]"
```

To enable the TRACE HTTP method, use:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomee/conf/server.xml':  
  {'connector.allowTrace':true}]"
```

## Configure the maximum number of active sessions

The SAP Java Buildpack provides the default configurations for unlimited sessions for the TomEE application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature. To limit the number of active sessions set the `maxActiveSessions` attribute on a `Manager` element, for 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 `web.xml`:

```
<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 using `app_context_root` in the application `manifest.yml` file. For example:

```
...
env:
  JBP_CONFIG_TOMCAT: "[tomee:{app_context_root: test_context_path}]"
...
```

### 3.1.9.1.3 TomEE 7

By default web applications pushed with the SAP Java buildpack are running in an Apache Tomcat container.

Applications could explicitly define the targeted application container - Apache TomEE 7, by using the `TARGET_RUNTIME` environment variable in the application `manifest.yml` file.

#### ↳ Sample Code

`manifest.yml`

```
---
applications:
- name: <APP_NAME>
  ...
env:
  TARGET_RUNTIME: tomee7
```

## Provided APIs

The tomee7 application runtime container provides the following standard APIs:

Runtime	TomEE 7	Supported Specification Version
tomee7	Apache TomEE 7 (Java EE 7 Web Profile)	Java Servlet 3.1 JavaServer Pages (JSP) 2.3 Expression Language (EL) 3.0 WebSocket 1.1 Standard Tag Library for JavaServer Pages (JSTL) 1.2 Java API for RESTful Web Services (JAX-RS) 2.0 JavaServer Faces (JSF) 2.2 Debugging Support for Other Languages 1.0 Enterprise JavaBeans (EJB) Lite 3.2 Java Transaction API (JTA) 1.2 Java Persistence API (JPA) 2.1 Bean Validation 1.1 Managed Beans 1.0 Interceptors 1.2 Common Annotations for Java Platform 1.2 Dependency Injection for Java 1.0 Contexts and Dependency Injection for Java EE platform 1.1

## Customizing the SAP Java Buildpack Defaults

The SAP Java Buildpack provides some default configurations for the Apache TomEE 7 application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature.

Below is a list with all of the placeholders which could 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 which can be used to enable or disable 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

The SAP Java Buildpack provides the default configurations for unlimited sessions for the TomEE 7 application container which could be customized by the application with the [Resource Configuration \[page 170\]](#) feature. To limit the number of active sessions set the `maxActiveSessions` attribute on a `Manager` element, for 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 `web.xml`:

```
<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 using `app_context_root` in the application `manifest.yml` file. For example:

```
...
env:
  JBP_CONFIG_TOMCAT: "[tomee7:{app_context_root: test_context_path}]"
...
```

### 3.1.9.1.4 Java Main

You can create a Java application that starts its own run time. 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 \[page 170\]](#) feature of the buildpack.

##### i Note

The resource configurations needed for the database connection are not applicable for the Java Main applications. For more information about database connections, see Related Information.

#### Context

In this section such applications will be referred to as Java Main applications. The application container provided by the SAP Java 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: p123456
Created-By: Apache Maven 3.3.3
Build-Jdk: 1.8.0_45
Main-Class: com.companya.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 fat 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.ServicesResourceTransformer" />
```

```

<transformer
implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTransformer">
    <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. Push the application to Cloud Foundry.

For the purpose use `cf push` command.

## Example

Donna Moore would like to create a Java Main application to use its own run time. For that purpose she performs the following steps:

- Creates a `sample_main` application. Using Spring Boot for that purpose.
- Navigates to the `sample_main` directory of the application, using the command line tool, and builds it. She can perform this operation with the command `mvn clean install`.
- After the build is successful, she checks that the `sample_main` directory of the application, contains a `sample_main.jar` file.
- She opens the `sample_main.jar` file and checks that the `META-INF/MANIFEST.MF` file contains the Main-Class header with value, the name of the main class.
- She adds the path to the JAR file in the `manifest.yml` file.

She needs to do that to make sure that the application can be pushed to the Cloud Foundry Environment. For that purpose, she adds the following line in the `manifest.yml` file:

```
path: ./target/sample_main.jar
```

- Finally, she pushes the Java Main application with the following command:

```
cf push sample_main
```

## Related Information

[Configuring a Database Connection \[page 182\]](#)

### 3.1.9.2 Customizing the Java Virtual Machine (JVM) Settings

#### 3.1.9.2.1 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: '[from_environment: false, java_opts: '-DtestJBPCconfig=%PATH% -DtestJBPCconfig1="test test" -DtestJBPCconfig2=%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 "[from_environment: false, java_opts: '-DtestJBPCconfig=%PATH% -DtestJBPCconfig1=\"test test\" -DtestJBPCconfig2=\"^%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 to 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: '[from_environment: false, java_opts: ''-
DtestJBPCConfig=\$PATH -DtestJBPCConfig1="test test" -
DtestJBPCConfig2="\$PATH"]'
```

### 3.1.9.2.2 Java Memory Assistant [AWS, Azure, or GCP Regions]

#### i Note

The content in this section is not relevant for China (Shanghai) and Government Cloud (US) regions.

A Java agent that generates heap dumps based on preconfigured conditions in the memory usage of the application. See [Java Memory Assistant](#).

When enabled in the buildpack, the agent will generate two files – \*.hprof (heap dump) and \*.addons, when the configured memory limits are met. The \*.addons file contains:

- command line parameters
- the implemented interfaces for the classes
- information (name) about transient fields for the classes
- a class and metaspace statistic
- stack traces of the last OOM errors

See [Java Memory Assistant Framework](#).

### 3.1.9.2.3 Java Out Of Memory Behavior

The SAP Java Buildpack prints a histogram of the heap to the logs, when the JVM encounters a terminal failure. In addition to that, if the application is bound to a volume service with name or tag that contains *heap-dump*, a heap dump file is also generated and stored in the mounted volume.

#### ↳ 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     |
[J
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. It is possible to deactivate it by adding the `-XX:`

`+ExitVMOnOutOfMemoryError` property (through the `JBP_CONFIG_JAVA_OPTS` property) in the application manifest file.

#### ↳ Sample Code

```
JBP_CONFIG_JAVA_OPTS: 'false, java_opts: ''-XX:+ExitVMOnOutOfMemoryError'''
```

For more information about the *jvmkill* agent, see [Cloud Foundry jvmkill documentation](#).

### 3.1.9.2.4 Using an Agent

You can use any agent with the SAP Java Build pack. The agent must be included in the `.jar` or `.war` archive of your application. The SAP Java Build pack extracts the agent when the application is deployed. You can check if the agent is extracted to the expected location by using the `cf ssh` command.

To use an agent with the SAP Java Build pack, set the `<JBP_CONFIG_JAVA_OPTS>` environment variable as shown in the following example:

```
env:  
  JBP_CONFIG_JAVA_OPTS: '[java_opts: "-javaagent:<PathToYourAgent>"]'
```

The Java agent is platform-agnostic, but must be compatible to the version of the JVM you are using.

You can also use a native agent. Since a native agent is a dynamic library, it must be compatible with the architecture of the platform. For Cloud Foundry, this is Linux x86\_64. As the application developer you must make sure that the correct agent is used and apply updates whenever they are needed. To use a native agent, set the `<JBP_CONFIG_JAVA_OPTS>` as follows:

```
env:  
  JBP_CONFIG_JAVA_OPTS: '[java_opts: "-agentpath:<PathToYourAgent>"]'
```

### 3.1.9.2.5 SapMachine

SapMachine  is an alternative to SAP JVM. It provides a Java Runtime Environment (JRE) with Java 11, while SAP JVM provides a JRE with Java 8. It works only with the [Tomcat \[page 149\]](#) and [Java Main \[page 157\]](#) application containers.

#### Activation

To activate SapMachine (instead of using the default SAP JVM) using the SAP Java Buildpack, you have to add the following environment variable:

```
---  
applications:  
- name: <app-name>  
  ...  
  env:  
    JBP_CONFIG_COMPONENTS: "jres:  
      ['com.sap.xs.java.buildpack.jdk.SAPMachineJDK']"  
  ...
```

#### Related Information

[Tomcat \[page 149\]](#)

[Java Main \[page 157\]](#)

<https://sap.github.io/SapMachine/> ↗

### 3.1.9.2.6 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.

The SAP Java Buildpack provides two options for a memory calculator:

- [Memory Calculator V2 \[page 164\]](#) - this memory calculator is activated by default.
- [Memory Calculator V1 \(SAP JVM Memory Calculator\) \[page 165\]](#) - this is an optional memory calculator and could be activated with the following environment variable:

```
---  
applications:  
- name: <app-name>  
...  
env:  
  MEMORY_CALCULATOR_V1: true  
...
```

#### 3.1.9.2.6.1 Memory Calculator V2

##### Customization

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']"
```

*memory\_calculator\_v2* section in [config/sapjvm.yml \[page 165\]](#) configuration file has two sizing options: *stack\_threads* (an estimate of the number of threads that will be used by the application) and *class\_count* (an estimate of the number of classes that will be loaded). You can customize those two memory options by using the JBP\_CONFIG\_SAPJVM environment variable:

```
---  
applications:  
- name: <app-name>  
...  
env:  
  JBP_CONFIG_SAPJVM: "[memory_calculator_v2: {stack_threads: 266, class_count:  
1001}]"
```

## Related Information

[Java Buildpack Memory Calculator](#)

### 3.1.9.2.6.2 Memory Calculator V1 (SAP JVM Memory Calculator)

#### 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 an 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 relative 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 *-Xms*,*-Xmx* JVM options
- *metaspace* - configure sizing options for the metaspace. Affects *-XX:MetaspaceSize*, *-XX:MaxMetaspaceSize* JVM options
- *stack* - configure sizing options for the stack. Affects *-Xss* JVM option
- *native* - serves to represent the rest of the memory (different to heap, stack, metaspace) in the calculations performed by the SAP JVM Memory Calculator. No JVM options are affected from 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 which 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% meaning that the memory calculation will result in *-Xms=-Xmx* and *-XX:MetaspaceSize=-XX:MaxMetaspaceSize*. If those values are set to 50% the *-Xmx = 2\*Xms* and *-XX:MaxMetaspaceSize=2\*XX:MetaspaceSize*.
- *memory\_settings* - for details see OutOfMemory error
- *memory\_calculator\_v2* - for details see Memory Calculator V2

## Customizing the Default Settings

There are two possible ways to customize the default settings - during application staging and during the 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 and all of them include using the `set-env` command either on XS advanced 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 which 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 say that the memory\_sizes configuration for heap is 10M..100M. 120M goes beyond the configured range. No matter than according to memory\_heuristics 120M could be allocated for heap the chosen value for heap size would be 100M. The calculation goes on until sizes are calculated for all of the regions - heap, metaspace, stack, native. Finally, the memory\_initials will be respected. This would affect the values for -Xms, -Xmx, -XX:MetaspaceSize, -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 have  $\text{estimated\_number\_of\_threads} = ((5/100)*1024M) = 51.2M$  space for stack. Considering 1M per thread (which is by default the -Xss size of the SAPJVM) we get  $\text{estimated\_number\_of\_threads} = 51.2$

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* /  $\text{estimated\_number\_of\_threads} = 51.2M / 51.2 = 1M$

## Example 2 (memory limit of 1G with customized settings of the memory calculator)

Memory limit	Memory calculator settings
1G	<i>memory_sizes:</i> metaspace: 64m..70m <i>memory_heuristics:</i> heap: 75 metaspace: 10, stack: 5 native: 10 <i>memory_initials:</i> heap: 100% metaspace: 50%

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 have estimated\_number\_of\_threads=((5/100)\*1024M) = 51.2M space for stack. Considering 1M per thread (which is by default the -Xss size of the SAPJVM) we 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*  
*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 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 is now  $(1G - 70M) = 954M$ .

The same calculation takes place, this time the metaspace is not considered because its 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* so 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 / estimated_number_of_threads = 53 / 51.2 = 1060K
```

## Related Information

[Memory Calculator V2 \[page 164\]](#)

### 3.1.9.3 Overriding Resources in Droplet

Applications can override resources in the droplet by placing files in `META-INF/sap_java_buildpack/resources/`. These files override files inside the droplet that have the same relative path to the droplet root.

The following example shows how application could override the default Tomcat's `server.xml`. Having the following structure in the application files:

```
META-INF/  
- sap_java_buildpack/  
  - resources/  
    - tomcat/  
      - conf/  
        - server.xml/  
WEB-INF/
```

will result in the following file structure in the droplet:

```
META-INF/  
- .sap_java_buildpack/  
  - tomcat  
    - conf/  
      - server.xml  
WEB-INF/
```

Meaning that when the Tomcat container is started the `server.xml` file used will be the one coming from the application.

This mechanism also allows of adding new resources inside of the droplet on their respective places based on where the files are located under `META-INF/sap_java_buildpack/resources/`.

### 3.1.9.4 Resource Configuration

Both application containers [Tomcat \[page 149\]](#) and [TomEE \[page 151\]](#) are configured through text configuration files, examples are the `conf/server.xml`, `conf/tome.xml` or `conf/logging.properties` ([Apache Tomcat 8 Configuration Reference ↗](#)). Resource configuration feature of the 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}). In order to prevent undesired modifications all files that could be changed must be explicitly listed by the application in a corresponding `resource_configuration.yml` file along with default values for all placeholders.

Example `resource_configuration.yml`:

```
---
filepath:
  key1: value1
  key2: value2
filepath2:
  key1: value1
```

The `filepath` must start with either `tomcat` or `tomee` 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:

```
<application_container>/webapps/ROOT/
<path_to_file_relative_to_the_root_of_the_application>
```

When changing parametrized values inside application container's configuration the file path should look like:

```
<application_container>/conf/<path_to_config_file>
```

Example:

```
tomcat/webapps/ROOT/WEB-INF/mytextfield.txt:
  placeholder: "defaultValue"
tomcat/conf/server.xml
  connector.maxThreads: 800
```

Example:

Given an application with the following application files:

```
src/
  main/
    java/
    resources/
    webapp/
      META-INF/
        sap_java_buildpack/
          config/
            resource_configuration.yml
      WEB-INF/
        mytextfield.txt
```

Where the content of `mytextfield.txt` is:

```
My hometown is ${hometown}.
```

The `resource_configuration.yml` looks like:

```
---
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.
```

The example below demonstrates how to provide values for placeholders during staging: Given the example above add the following environment variable in the `manifest.yml` file of the application:

```
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 provided in the example above defaults. Staging the application with this environment variable will modify the content of `mytextfield.txt` to:

```
My hometown is Paris.
```

### 3.1.9.5 Context Root Redirect

The SAP Java Buildpack provides context root redirect functionality.

When you call a web application without adding its runtime ([Tomcat \[page 149\]](#), [TomEE \[page 151\]](#) or [TomEE 7 \[page 154\]](#)) context path to the URL, it will be automatically appended.

Example:

If you have configured as context path `/test_context_path` and the Web application is available on `/test_app`, when you call:

```
<HOST>:<PORT>/test_app
```

you'll be redirected to:

```
<HOST>:<PORT>/test_context_path/test_app
```

The default context path value for Tomcat, TomEE, and TomEE 7 is "" (Empty String). For details how to change this default value check here: tomcat[Tomcat \[page 149\]](#), tomee[TomEE \[page 151\]](#), tomee7[TomEE 7 \[page 154\]](#)

### 3.1.9.6 Logging and Tracing

#### Context

SAP Java Buildpack integrates the <https://github.com/SAP/cf-java-logging-support> libraries so that applications could produce logs in the JSON format that could be parsed by Kibana out of the box.

## Procedure

- Bind the application to the logging service.

In order to produce application logs and forward them to the Kibana (ELK) stack, the application should be bound to the application logging service.

- Add the `com.sap.hcp.cf.logging.servlet.filter.RequestLoggingFilter` filter to the `web.xml`, if request metrics should be generated for applications using the Tomcat/TomEE application containers.

```
<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>
```

### i Note

The default logging level of `com.sap.hcp.cf.logging.servlet.filter.RequestLoggingFilter` is set to `INFO`.

- Check the logs of the application.

Use the `cf logs` command to get the logs of the application.

```
cf logs myapp --recent
```

- Change the logging level of a logging location.

Applications could change the logging level of a specific location by setting the `SET_LOGGING_LEVEL` environment variable in the `manifest.yml` file of the application.

```
env:
  SET_LOGGING_LEVEL: '{com.sap.sample.java.LogInfo: INFO,
  com.sap.sample.java.LogWarn: WARN}'
```

## Related Information

[Configure a Java Application for Logs and Traces \[page 174\]](#)

### 3.1.9.6.1 Configure a Java Application for Logs and Traces

Configure the collection of log and trace messages generated by a Java application in the Cloud Foundry Environment.

#### Prerequisites

- You have installed and configured Maven.
- You use SAP JVM 8 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 runtimes provided by the SAP Java buildpack. Packing them in the application could cause problems during class loading.

#### ℹ Note

It is the application's responsibility to ensure that logback is configured in a secure way in the cases when the application overrides the default logback configuration, included in the SAP Java buildpack, for example the logback appenders.

#### 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](#) class. You can use it to configure your Java application to generate logs and traces, and if appropriate set the logging or tracing level.

#### Procedure

1. Instruct Maven that the application should not package the SLF4J dependency.

This dependency is already provided by the runtime, use the scope provided in order not to pack the slf4j-api in your application.

```
<dependency>
  <groupId>org.slf4j</groupId>
  <artifactId>slf4j-api</artifactId>
  <version>1.7.12</version>
  <scope>provided</scope>
</dependency>
```

2. Import the SLF4J classes into the application code ([org.slf4j.Logger](#) and [org.slf4j.LoggerFactory](#)) and develop the log handling code.

## Example

### ↳ Sample Code

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
public void logging () {
    final Logger logger =
LoggerFactory.getLogger(Log4JServlet.class);
    logger.debug("this is record logged through SLF4J param1={}, param2={}.", "value1", 1);
    logger.info("This is slf4j info");
}
```

## Related Information

### 3.1.9.6.2 Configure the Access Logs of an Application

The SAP Java Buildpack uses the [logback-access](#) module to provide HTTP-access log functionality.

## Prerequisites

- [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

## 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 is not SAP-specific either. The default pattern defined in configuration file is "%date %r %s %b".

In order to find the access log files of your application, you have to connect via SSH to the Cloud Foundry container and go to the `logs` directory.

## Procedure

1. Modify the default configuration.

SAP Java buildpack uses LogbackValve to configure access logs in Tomcat and TomEE. There are four environment variables (`<ACCESS_LOG_FILE>`, `<ACCESS_LOG_FILE_COUNT>`, `<ACCESS_LOG_FILE_MAX_SIZE>`, `<logback.access.log.xs.pattern>`) which are used in the configuration file and could be overwritten.

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 which contains those file and the name of the files, with the environment variable `<ACCESS_LOG_FILE>`. In order to do it you have to define the full path to the desired directory and name of the file.

```
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.

```
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.

```
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. The SAP Java buildpack uses the [Resource configuration](#) functionality to allow this. Currently the change of the `pattern` attribute in the configuration file of Tomcat and TomEE is supported. The application has to provide the new value via the JBP\_CONFIG mechanism:

- Changing the format of the access log for Tomcat:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomcat/conf/logback-access-  
localhost.xml':{'logback.access.log.xs.pattern' : 'combined'}]"
```

- Changing the format of the access log for TomEE:

```
env:  
  JBP_CONFIG_RESOURCE_CONFIGURATION: "['tomee/conf/logback-access-  
localhost.xml':{'logback.access.log.xs.pattern' : 'combined'}]"
```

## 3.1.9.7 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.

- Debug an Application Running on SAP JVM [page 177]
- Debug an Application Running on Java SE [page 179]

### 3.1.9.7.1 Debug an Application Running on SAP JVM

You can debug an application running on a Cloud Foundry container that is using SAP JVM. By using SAP JVM, you can enable debugging on-demand without having to restart the application or the JVM.

#### Prerequisites

- Download and install the Cloud Foundry Command Line Interface (cf CLI) and log on to Cloud Foundry. See [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- Deploy your application using the SAP Java Buildpack. From the cf CLI, execute `cf push <app name> -p <my war file> -b sap_java_buildpack`.
- Ensure that SSH is enabled for the application. See <https://docs.cloudfoundry.org/devguide/deploy-apps/ssh-apps.html>

#### i Note

SAP JVM is included in the SAP Java Buildpack. With SAP JVM, you can enable debugging on-demand. You do not need to set any debugging parameters.

#### Context

After enabling the debugging port, you need to open an SSH tunnel, which connects to that 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

### i Note

The default port is 8000.

4. To exit `jvmmon`, execute `q`.
5. From the cf CLI, open the SSH tunnel by executing `cf ssh <app name> -N -T -L 8000:127.0.0.1:8000`.
6. Connect a Java debugger to your application. For example, use the standard Java debugger provided by Eclipse IDE and connect to `localhost:8000`.

## 3.1.9.7.2 Debug a Java Web Application Running on SAPMachine

Debug an application running on a Cloud Foundry container that is using SAPMachine.

### Prerequisites

Download and Install the Cloud Foundry Command Line Interface (cf CLI) and log on to Cloud Foundry. See [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

### Context

To debug an application 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. By default the `agentlib`, which enables the debug 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,onjcmd=y']"
```

You may change the default port 8000 in the command.

2. To enable debugging or to check the debugging state of your JVM, run `j cmd` in your Cloud Foundry container by executing:

```
cf ssh <APPLICATION_NAME> -c 'export JAVA_PID=`ps -C java -o pid=` && app/META-INF/.sap_java_buildpack/sap_machine_jdk/bin/jcmd $JAVA_PID VM.start_java_debugging'
```

The following is an example of the information displayed by `j cmd`:

```
<pid>:  
Debugging has been started.  
Transport : dt_socket  
Address   : 8000
```

### i Note

The default port is 8000.

3. To enable SSH tunneling in your application, execute `cf enable-ssh`.
4. To open the SSH tunnel, execute `cf ssh -N -T -L 8000:127.0.0.1:8000`. Your local port 8000 is connected to the debugging port 8000 of the JVM, which running in the Cloud Foundry container.

### ⚠ Caution

The connection is active until you close the SSH tunnel. After you have finished debugging, close the SSH tunnel by pressing `Ctrl` + `C`. Connect a Java debugger to your application. For example, use the standard Java debugger provided by Eclipse IDE and connect to `localhost:8000`.

## 3.1.9.7.3 Debug an Application Running on Java SE

You can debug an application running on a Cloud Foundry container that is using the standard Java community build pack.

### Prerequisites

Download and install the Cloud Foundry Command Line Interface (cf CLI) and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

## Context

To debug an application 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.

### i Note

The default port is 8000.

### ⚠ Caution

The connection is active until you close the SSH tunnel. After you have finished 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`.

## 3.1.9.8 Dynatrace Integration

## Context

Dynatrace OneAgent is a Java agent that sends all captured monitoring data to your monitoring environment for analysis. The monitoring environment for Cloud Foundry environment resides in the Dynatrace cloud monitoring environment, see [How Do I Monitor Cloud Foundry Applications](#).

## Procedure

1. Obtain an environment ID and Token.

Follow the steps described in the Generate PaaS Token section in [How Do I Monitor Cloud Foundry Applications](#).

- Create a user provided service.

The service name contains the string dynatrace (for example - `dynatrace-service`) where `environmentid` and `apitoken` are the ones generated in the previous step. The `apiurl` should be set to `https://<YourDynatraceServerURL>/e/<environmentid>/api`.

```
cf cups dynatrace-service -p "environmentid, apitoken, apiurl"
```

For more details on how to create the user provided service refer to Option 1: Create a user-provided service in the Create a Dynatrace service in your Cloud Foundry Environment section in [How Do I Monitor Cloud Foundry Applications](#).

- Bind the application to the service instance created in the previous step, using the `manifest.yml` file.

#### ↳ Sample Code

`manifest.yml`

```
---
applications:
- name: myapp
  memory: 1G
  instances: 1
  services:
    - dynatrace-service
```

### 3.1.9.9 Multiple Buildpack Framework

The Multiple Buildpack Framework enables the 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.

#### i Note

If you try to use the SAP Java Buildpack as a non-final buildpack, an error will be thrown.

When the 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.

## 3.1.9.10 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 \[page 149\]](#), [TomEE \[page 151\]](#) and [TomEE 7 \[page 154\]](#) application containers.

- [Configure a Database Connection for the Tomcat Application Container \[page 182\]](#)
- [Configure a Database Connection for the TomEE Application Container \[page 183\]](#)
- [Database Connection Configuration Details \[page 186\]](#)
- [SAP HANA HDI Data Source Usage \[page 188\]](#)

### 3.1.9.10.1 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.

### 3.1.9.10.2 Configure a Database Connection for the TomEE 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.tomee.TomEEDataSourceFactory"
    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.

```
---
tomee/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: "[{'tomee/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.

### 3.1.9.10.3 Configure a Database Connection for the TomEE 7 Application Container

#### Procedure

1. Create a `resources.xml` file.

##### i 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 \[page 188\]](#)

[Database Connection Configuration Details \[page 186\]](#)

[Configuring a Database Connection \[page 182\]](#)

### 3.1.9.10.4 Database Connection Configuration Details

Define details of the database connection used by your Java Web Application running on Cloud Foundry Environment with the 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 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 \[page 170\]](#) 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 \[page 170\]](#).

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' }]"
```

## 3.1.9.10.5 SAP HANA HDI Data Source Usage

### 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 file.

#### ↳ Sample Code

`manifest.yml`

```
services:
- my-hdi-container
```

3. Add the resource reference to the `web.xml` file, which must have the name of the service instance.

#### ↳ Sample Code

web.xml

```
<resource-ref>
    <res-ref-name>jdbc/my-hdi-container</res-ref-name>
    <res-type>javax.sql.DataSource</res-type>
</resource-ref>
```

4. Look up the data source in your code.

You can find the reference to the data source in the following ways:

- 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");
```

### 3.1.9.11 SAP Java Connector

The SAP Java buildpack provides an option to use the SAP Java Connector.

#### Activation

The SAP Java buildpack provides an option to use the [SAP Java Connector \(SAP JCo\)](#).

To activate SAP JCo in the SAP Java buildpack, the application needs to be bound to connectivity and destination services, see [Create and Bind a Connectivity Service Instance](#) and [Create and Bind a Destination Service Instance](#):

#### ↳ Sample Code

manifest.yml

```
---
applications:
- name: <APP_NAME>
  ...
services:
  - connection_service_instance
  - destination_service_instance
```

The activation of SAP JCo will provide all relevant libraries in the application container, so the [Java Connector API About JCocan](#) be used.

## Limitations

It is not possible to use the SAP Java Connector with Spring Boot applications.

## Related Information

[Invoking ABAP Function Modules via RFC](#)

Tutorial: [Invoke ABAP Function Modules in On-Premise ABAP Systems](#)

### 3.1.9.12 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 stand-alone 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 \[page 191\]](#).

Open a debugging connection using an SSH tunnel. For more information, see [Debug an Application Running on SAP JVM \[page 177\]](#)

#### 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*.

**i Note**

Your port number is your local SSH tunnel endpoint.

4. Choose the analysis you want to run and specify your profiling parameters.

**i 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.

### 3.1.9.12.1 Install the SAP JVM Tools in Eclipse

The SAP JVM Profiler is included in the SAP JVM Tools.

#### Procedure

1. Open Eclipse and go to *Help* *Install New Software*.
2. In the *Work with* combo box enter <https://tools.hana.ondemand.com/oxygen> and choose *SAP Cloud Platform Tools* *SAP JVM Tools*.
3. Choose *Next* and follow the installation wizard.

### 3.1.9.12.2 (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 for Eclipse. For more information, see [Install the SAP JVM Tools in Eclipse \[page 191\]](#).

Download and install the Cloud Foundry Command Line Interface (cf CLI) and log on to Cloud Foundry. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

## 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 Eclipse. Go to  [Help](#) > [Help Contents](#) > [SAP JVM Profiler](#).

### i 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 Eclipse, 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`.

Your application is added to the VM Explorer.

### 3.1.9.13 Bill of Materials (BOM)

The versions of the SAP Java buildpack dependencies and the provided APIs from supported runtime containers, could be consumed through a Bill Of Materials (BOM). The BOM can be used to control the versions of a project's dependencies.

## Usage

There are three options when using a BOM depending on the application runtime - Tomcat, TomEE, TomEE7.

### i Note

The versions of BOM artifacts are related to the versions of the SAP Java buildpack. For example, if your application uses SAP Java buildpack version 1.9.1, then you have to use BOM version 1.9.1 as well.

Use the `dependencyManagement` tag to add the desired BOM artifact.

- For Tomcat:

```
...
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.sap.cloud.sjb</groupId>
      <artifactId>cf-tomcat-bom</artifactId>
      <version>1.9.1</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
...
```

- For TomEE:

```
...
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.sap.cloud.sjb</groupId>
      <artifactId>cf-tomee-bom</artifactId>
      <version>1.9.1</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
...
```

- For TomEE 7:

```
...
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>com.sap.cloud.sjb</groupId>
      <artifactId>cf-tomee7-bom</artifactId>
      <version>1.9.1</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>
...
```

Use *groupId* and *artifactId* to add the dependencies that you want.

```
...
<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>
...

```

### 3.1.10 Developing Multitenant Applications in the Cloud Foundry Environment

In the Cloud Foundry environment, as an application provider, you can develop and run multitenant applications, and share them with multiple consumers (tenants) simultaneously on SAP Cloud Platform. Each consumer accesses the application through a dedicated URL that identifies their tenant.

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

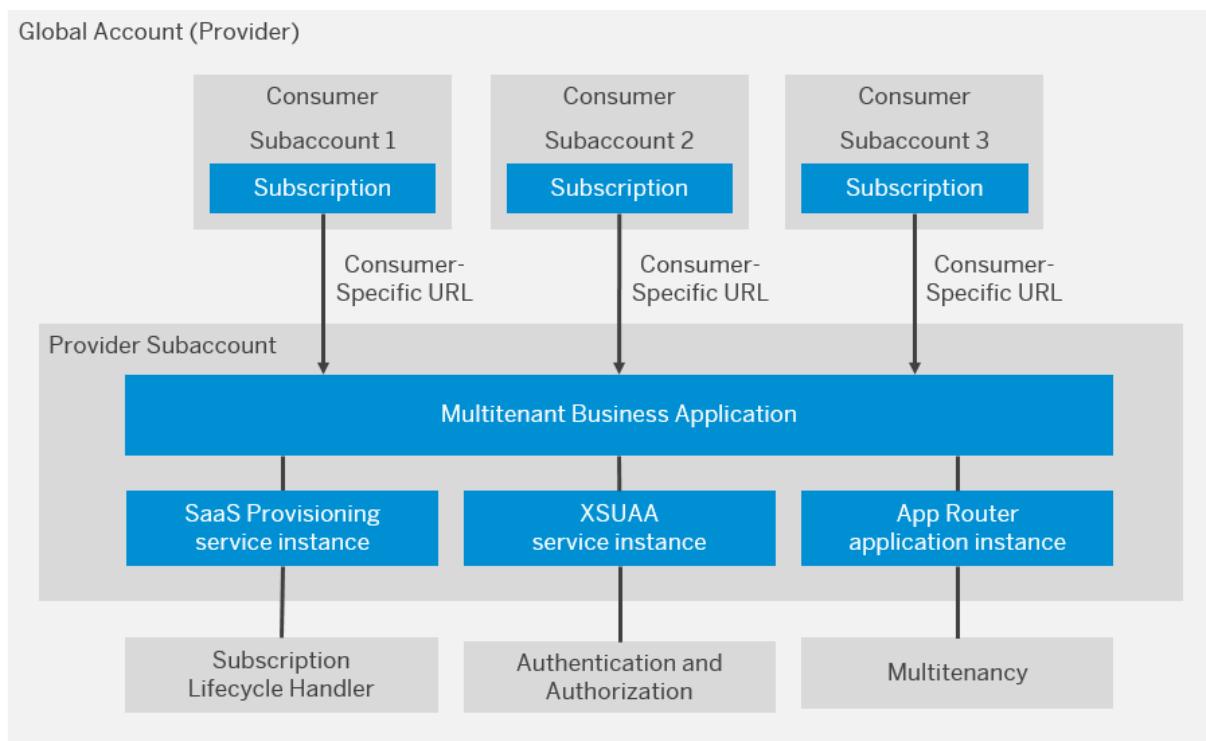
For a consumer to use a tenant-aware application on SAP Cloud Platform, the application owner must ensure that each consumer:

1. Has a dedicated subaccount in the application provider's global account.
2. Subscribes to the application in the cockpit.  
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

For more information about these consumer-related steps, see the [Getting Started with Multitenant Application Subscriptions in the Cloud Foundry Environment \[page 29\]](#).

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 registered with the SaaS Provisioning (saas-registry) service (with the application plan) to make the application available for subscription to consumers.



This section describes the workflow and requirements for developing and deploying a multitenant application in the Cloud Foundry environment.

- [1. Develop the Multitenant Application \[page 196\]](#)  
In the Cloud Foundry environment of SAP Cloud Platform, 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 \[page 199\]](#)  
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 \[page 200\]](#)  
To authenticate business users of the application at runtime, use the tenant-aware approuter application and xsuaa service in SAP Cloud Platform.
- [4. Bind the Multitenant Application and Application Router to the xsuaa Service Instance \[page 203\]](#)  
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 SaaS Provisioning Service \[page 204\]](#)  
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 (`saas-registry`).

## Related Information

[Using SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 208\]](#)

### 3.1.10.1 Develop the Multitenant Application

In the Cloud Foundry environment of SAP Cloud Platform, 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 `SaaS 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.

#### → Tip

Beginner and novice developers can use this [sample Hello World application](#) and read this [blog about SaaS provisioning](#) to understand the basics of developing and deploying a multitenant application in the Cloud Foundry environment. Then, you can proceed to this more technical [sample application](#) and supplementary [blog](#), which cover additional components and can help ensure your multitenant application is compliant with the requirements mentioned above.

Both sample applications are published in Github, and include instructions and code samples.

#### Procedure

1. Develop the application as for any other Java and Node.js application on the SAP Cloud Platform.  
For more information, see [Developing Java in the Cloud Foundry Environment \[page 146\]](#).
2. In your application, you must implement a REST API that will be called by the `SaaS Provisioning (saas-registry)` service 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 `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 \[page 200\]](#))

#### 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
    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
    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 SaaS Provisioning Service \[page 204\]](#).

- Configure the application's authentication and authorization for the SAP HANA XS advanced Java run time.

For general instructions, see [SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#) and [Configuring Authentication for Spring Boot Applications \[page 1117\]](#).

- 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 SaaS Provisioning Service (`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)"
      ]
    },
    ...
  ],
  ...
}
```

For more information, see [The Application Security Descriptor \[page 253\]](#) and [Granting Scope Access to Another Application \[page 282\]](#).

- 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 \[page 199\]](#)), 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 a Service Instance from the xsuaa Service \[page 278\]](#) and [Create Spaces Using the Cloud Foundry Command Line Interface \[page 965\]](#).

## 3.1.10.2 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 \[Feature Set A\] \[page 762\]](#).

2. In the subaccount, create a space in a Cloud Foundry organization.  
For general instructions, see [Create Spaces \[page 922\]](#).
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 \[page 781\]](#).
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 \[page 62\]](#).

### 3.1.10.3 Configure the approuter Application

To authenticate business users of the application at runtime, use the tenant-aware `approuter` application and `xsuaa` service in SAP Cloud Platform.

#### 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 \[page 987\]](#)

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 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. 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 in the Cloud Foundry Environment \[page 1104\]](#)
- [What Is Authorization and Trust Management? \[page 1096\]](#)
- [Authentication for Applications \[page 1115\]](#)
- [Configuring Authentication for Spring Boot Applications \[page 1117\]](#)

## Procedure

1. Create the application structure of the application router and configure it accordingly.

For general instructions, see [Application Router \[page 77\]](#).

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 \[page 111\]](#).

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 \[page 86\]](#).

4. Use the `push` command in the cf CLI to deploy the `approuter` application to the Cloud Foundry space where your multitenant is deployed.

### 3.1.10.4 Bind the Multitenant Application and Application Router to the xsuaa Service Instance

Bind your multitenant application and the `approuter` application to the `xsuaa` service 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 196]</a> .
<code>APPROUTER_APP_NAME</code>	The name of the <code>xsuaa</code> application you deployed in <a href="#">Configure the approuter Application [page 200]</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 cockpit.

For general instructions, see [Bind the xsuaa Service Instance to the Application \[page 280\]](#).

### 3.1.10.5 Register the Multitenant Application to the 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 (`saaS-registry`).

#### Context

The SaaS Provisioning service allows application providers to register multitenant applications and services in the Cloud Foundry environment in SAP Cloud Platform. 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.
  - 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>"  
    },  
    "displayName" : "<displayname>",  
    "description" : "<description>",  
    "category" : "<category>"  
}
```

Specify the following parameters:

Parameters	Description
<code>xsappname</code>	The <code>xsappname</code> configured in the security descriptor file used to create the XSUAA instance (see <a href="#">Develop the Multitenant Application [page 196]</a> ).
<code>getDependencies</code>	(Optional) 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.

**i Note**

The JSON response of the callback must be encoded as either UTF8, UTF16, or UTF32, otherwise an error is returned.

Parameters	Description
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).
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.

### ↳ 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"
}
```

- b. 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.
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
```

#### → Tip

You can create the SaaS Provisioning service instance directly in the cockpit.

For general instructions, see [Creating Service Instances \[page 372\]](#).

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"}]'
```

3. Bind the SaaS Provisioning service instance that you just 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
<code>APP_NAME</code>	The ID of your deployed multitenant application.
<code>SAAS_REGISTRY_SERVICE_INSTANCE</code>	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 cockpit.

For general instructions, see [Binding Service Instances to Applications \[page 376\]](#).

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
<code>APP_NAME</code>	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 in the Cloud Foundry environment 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 [Getting Started with Multitenant Application Subscriptions in the Cloud Foundry Environment \[page 29\]](#).

### 3.1.10.5.1 Unregister a Multitenant Application from the 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 following API to get the application subscriptions: [Get Application Subscriptions \[page 218\]](#).

In case there are subscribed tenants, unsubscribe them first. See: [Unsubscribe the Tenant from an Application \[page 216\]](#).

#### Context

The procedure described in this section uses the cf CLI.

→ Tip

You can delete a service instance directly in the 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
```

i 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.

## 3.1.10.6 Using SaaS Provisioning Service APIs to Manage Multitenant Applications

You can use the SaaS Provisioning service (`saas-registry`) APIs to manage your multitenant application:

- [Get Application Registration Details \[page 211\]](#)
- [Subscribe the Tenant to an Application \[page 213\]](#)
- [Unsubscribe the Tenant from an Application \[page 216\]](#)
- [Get Application Subscriptions \[page 218\]](#)
- [Update Subscription Dependencies \[page 221\]](#)
- [Get Subscription Job Information \[page 223\]](#)

## Prerequisites

You have obtained your application's JSON Web Token (JWT) from the `xsuaa` service instance by calling the [Get an Application Access Token \[page 209\]](#) API.

### 3.1.10.6.1 Get an Application Access Token

Use this API to get the application access token from the xsuaa service instance.

The token is a JSON Web Token (JWT).

For more information, see [JSON Web Token \(JWT\)](#).

You use this token to manage the SaaS Provisioning service APIs.

#### Request

**URI:** <"saas-registry"."credentials"."url">/oauth/token

**HTTP Method:** *POST*

#### Request Headers

Header	Required	Values
Content-Type	Yes	<application/x-www-form-urlencoded>
Authorization	Yes	Basic <basic authentication value>

#### 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 client_credentials.	Authorization protocol
client_id	Yes	String	The ID of the client associated with the SaaS Provisioning service instance.	Relative URL path or JavaScript source code

#### Request Example

```
POST /oauth/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
Authorization: Basic c2FtcGx1Y2xpZW50OnNhbXBsZXBhc3NvcmQ=
grant_type=client_credentials&client_id=sampleclient
```

## 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.
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.
ext_attr	JSON object	Embedded JSON object that contains information about the provider of the token and the ID of the service instance.
enhancer	String	Service that enhances the token with additional information.
serviceinstanceid	String	ID of the SaaS Provisioning service (saas-registry) instance.
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": "eyJhbGciOiJSUzI1NiIsImprdsI6Imh0d...",
  "token_type": "bearer",
  "expires_in": 43199,
  "scope": "uaa.resource lps-registry-broker!b4.job.read lps-registry-broker!b4.entitlement.read lps-registry-broker!b4.subscription.write lps-registry-broker!b4.subscription.read",
  "ext_attr": {
    "enhancer": "xsuaa",
```

```
        "serviceinstanceid": """6f823c94-abbd-4f4a-8u4e-873ffjUe7b20f"  
    },  
    "jti": "df6cb84439a541fab33d5b7c298debe1"  
}
```

## Related Information

[Using SaaS Provisioning Service APIs to Manage Multitenant Applications \[page 208\]](#)

### 3.1.10.6.2 Get Application Registration Details

Get registration details for a multitenant application from the SaaS Provisioning Service.

## Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

## Request

**URI:** <"saas-registry"."credentials"."saas\_registry\_url">/saas-manager/v1/application

**HTTP Method:** *GET*

### Request Headers

Header	Required	Values
Authorization	Yes	Bearer <application JWT>

### Request Parameters

None

### Request Example

```
GET saas-manager/v1/application HTTP/1.1  
Authorization: eyJhbGciOiJSUzI1NiIs...
```

## Response

Returns registration details for the application.

Format: [JSON](#)

### Response Status and Error Codes

Code	Description
200	Successfully returned
204	No content to show
401	Invalid JWT
500	Internal server error

### Response Headers

Header	Values
Content-Type	application/json; charset=UTF-8

### Response Properties

Property Name	Property Type	Description
serviceInstanceId	String	The ID of the multitenant application that is registered to the SaaS Provisioning service registry.
OrganizationGuid	String	The unique ID of the Cloud Foundry org where the app provider has deployed and registered the multitenant application.
spaceGuid	String	The unique ID of the Cloud Foundry space where the app provider has deployed and registered the multitenant application.
xsappname	String	The xsappname configured in the security descriptor file used to create the xsuaa service instance for the multitenant application.
appId	String	The ID returned by an xsuaa service instance after the app provider has connected the multitenant application to an xsuaa service instance.
appName	String	The unique registration name of the deployed multitenant application as defined by the app developer.
appUrls	URL	Any callback URLs that the multitenant application exposes.

Property Name	Property Type	Description
providerTenantId	String	The unique ID of the tenant that provides the multitenant application.
appType	String	The plan used to register the multitenant application or reusable service: <ul style="list-style-type: none"><li>• saasApplication: Registered entity is a multitenant application.</li><li>• saasService: Registered entity is a reuse service.</li></ul>
displayName	String	The display name of the application for customer-facing UIs.
description	String	The description of the multitenant application for customer-facing UIs.
category	String	The category to which the application is grouped in the <i>Subscriptions</i> page in the cockpit. If left empty, it gets assigned to the default category.
globalAccountId	String	ID of the global account where the app is hosted.

### Response Example

#### « Sample Code

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
{
  "serviceInstanceId": "7c1e5e62-2cba-40f1-a58a-338ce2903fc5",
  "organizationGuid": "cb69e279-d145-4200-8051-f3d3ea1933fa",
  "spaceGuid": "12a3de89-6661-499f-a0ea-43f24ac23d4e",
  "xsappname": "sample-saas-application-test",
  "appId": "sample-saas-application-test!t220",
  "appName": "6542fe75-2700-4401-be9a-8cb845a5dfa0",
  "appUrls": "{\"getDependencies\":\"http://localhost:8080/callback/v1.0/dependencies\", \"onSubscription\":\"http://localhost:8080.cfapps.stagingaws.hanavlab.ondemand.com/callback/v1.0/tenants/{tenantId}\"}",
  "providerTenantId": "4543c52d-3fa5-4ba5-8b22-7306d43abc94",
  "appType": "saasApplication",
  "displayName": "Sample saas app test",
  "description": "Sample multitenant application",
  "category": "SaaS",
  "globalAccountId": "saas-ga-demo"
}

```

### 3.1.10.6.3 Subscribe the Tenant to an Application

Subscribe a consumer subaccount tenant to a multitenant application.

This REST call is asynchronous.

After you execute the API, a subscription job is created.

You can only subscribe tenants that are in the same global account as the provider of the application.

Note that you can subscribe tenants that have the `SUBSCRIBE_FAILED` or `UNSUBSCRIBE_FAILED` status.

## Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

## Request

**URI:** <"saas-registry"."credentials"."saas\_registry\_url">/saas-manager/v1/application/tenants/<tenantId>/subscriptions

### URI Path Parameter

Parameter Name	Parameter Type	Description
tenantId	String	The ID of the tenant to subscribe

**HTTP Method:** `POST`

### Request Headers

Header	Required	Value
Authorization	Yes	Bearer <application JWT>
Content-Type	Yes	<application/json>

### Request Parameters

None

#### i Note

You must include an empty JSON object `{}` in the request body.

### Request Example

```
POST saas-manager/v1/application/tenants/900e2f70-95f2-40d3-a32a-eaae5203cfcc6/
subscriptions HTTP/1.1
Authorization: eyJhbGciOiJSUzI1NiIs...
Content-Type: application/json
{ }
```

## Response

Creates a subscription job.

**Content Type:** *TEXT*

### Response Status and Error Codes

Code	Description
202	Subscription process started.
400	Tenant doesn't exist. Reasons: <ul style="list-style-type: none"><li>• Invalid request body</li><li>• Subscription exists in status:<ul style="list-style-type: none"><li>◦ IN_PROCESS</li><li>◦ SUBSCRIBED</li><li>◦ UPDATE_FAILED</li></ul></li></ul>
401	Invalid JWT.
403	Authorization refused. Consumer tenant is not in the same global account as the provider tenant.
500	Internal server error.

### Response Headers

Header	Values
Content-Type	<text/plain; charset=UTF-8>
Location	<path to job status>

#### i Note

The <path to job status> serves as the relative path for the base request URI of the [Get Subscription Job Information \[page 223\]](#) API.

Use the Get Subscription Job Information API to query the status of the subscription job you created by calling the current API.

**Response Payload Content:** Job for the subscription of application: <`appId`> and tenant: <`tenantId`> was created.

Where:

- <`appId`> is the ID of the application to which you subscribed the tenant.
- <`tenantId`> is the ID of the subscribed tenant.

### Response Example

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
```

```
Location: /api/v2.0/jobs/afd10d2d-f235-4625-817d-cc2acac90de4
Job for subscription of application: 6542fe75-2700-4401-be9a-8cb845a5dfa0 and
tenant: 900e2f70-95f2-40d3-a32a-eaae5203cfc6, was created
```

### 3.1.10.6.4 Unsubscribe the Tenant from an Application

Unsubscribe a consumer subaccount tenant from a multitenant application.

This REST call is asynchronous.

After executing the API, a job to unsubscribe the tenant is created.

You can unsubscribe tenants that are either in the same global account, or in a different global account as the provider of the application.

You can unsubscribe tenants with the following statuses:

- SUBSCRIBED
- SUBSCRIBE\_FAILED
- UNSUBSCRIBE\_FAILED
- UPDATE\_FAILED

## Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

## Request

**URI:** <"saas-registry"."credentials"."saas\_registry\_url">/saas-manager/v1/application/tenants/<tenantId>/subscriptions

### URI Path Parameter

Parameter Name	Parameter Type	Description
tenantId	String	The ID of the tenant to unsubscribe

**HTTP Method:** `DELETE`

### Request Headers

Header	Required	Value
Authorization	Yes	Bearer <application JWT>
Content-Type	Yes	<application/json>

## Request Example

```
DELETE saas-manager/v1/application/tenants/900e2f70-95f2-40d3-a32a-eaae5203cf6/
subscriptions HTTP/1.1
Authorization: eyJhbGciOiJSUzI1NiI...
Content-Type: application/json
```

## Response

Creates a job to unsubscribe the tenant from an application.

**Content Type:** *TEXT*

### Response Status and Error Codes

Code	Description
202	Process to unsubscribe started.
400	Tenant not found. Reasons: <ul style="list-style-type: none"><li>Subscription exists in status:<ul style="list-style-type: none"><li>IN_PROCESS</li></ul></li><li>Invalid path parameter</li></ul>
401	Invalid JWT.
500	Internal server error.

### Response Headers

Header	Values
Content-Type	<text/plain; charset=UTF-8>
Location	<path to job status>

#### i Note

The <path to job status> serves as the relative path for the base request URI of the [Get Subscription Job Information \[page 223\]](#) API.

Use the Get Subscription Job Information API to query the status of the unsubscribe job you created by calling the current API.

**Response Payload Content:** Job to delete subscription for application with instance id: <`appId`> and consumer tenant id: <`tenantId`> was created.

Where:

- <`appId`> is the ID of the application from which you unsubscribed the tenant.
- <`tenantId`> is the ID of the unsubscribed tenant.

## Response Example

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Location: /api/v2.0/jobs/afd10d2d-f235-4625-817d-cc2acac90de4
Job to delete subscription for application with instance id: 7c1e5e62-2cba-40f1-
a58a-338ce2903fc5 and consumer tenant id: 900e2f70-95f2-40d3-a32a-eaae5203cf6
created
```

## 3.1.10.6.5 Get Application Subscriptions

Run this API to get information about the subscriptions of a multitenant application. The information you get includes the ID of the subscribed tenant, the subscription code, and subscription dependencies.

### Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

### Request

**URI:** <"saas-registry"."credentials"."saas\_registry\_url">/saas-manager/v1/application/subscriptions

#### URI Query Parameters

Parameter Name	Required	Parameter Type	Description
Authorization	Yes	string	For more information, see <a href="#">[page 209]</a>
tenantId	No	String	Get subscriptions by tenant ID.
state		String	Get subscriptions by state. Values: <ul style="list-style-type: none"><li>• IN_PROCESS</li><li>• SUBSCRIBED</li><li>• SUBSCRIBE_FAILED</li><li>• UNSUBSCRIBE_FAILED</li><li>• UPDATE_FAILED</li></ul>
globalAccountId	No	string	Get subscriptions by associated global account ID.

**HTTP Method:** [GET](#)

## Request Headers

Header	Required	Values
Authorization	Yes	Bearer <application JWT>
Content-Type	Yes	<application/json>

## Request Example

```
GET /saas-manager/v1/application/subscriptions/tenantId HTTP/1.1
Authorization: eyJhbGciOiJSUzI1NiIs...
Content-Type: application/json
```

## Response

Returns information about the subscriptions of a multitenant application.

Content Type:[JSON](#)

### Response Headers

Header	Values
content-type	application/json; charset=UTF-8

### Response Status and Error Codes

Code	Reason
200	Found subscriptions
400	Invalid parameters. Reasons: <ul style="list-style-type: none"><li>• Wrong tenant ID.</li><li>• Wrong tenant state. See the <b>URI Path Parameters</b> section for possible states.</li></ul>
401	Invalid JWT
500	Internal server error

### Response Properties

Property Name	Property Type	Description
subscriptions	Array	List of the tenants subscribed to the application.
url	URL	Application URL.
appName	String	The unique registration name of the deployed multitenant application, as defined by the app developer.

Property Name	Property Type	Description
consumerTenantId	String	Tenant ID of the global account or sub-account of the consumer that has subscribed to the multitenant application.
state	String	<p>State of the subscriptions.</p> <p>Possible states:</p> <ul style="list-style-type: none"> <li>• IN_PROCESS</li> <li>• SUBSCRIBED</li> <li>• SUBSCRIBE_FAILED</li> <li>• UPDATE_FAILED</li> </ul>
dependencies	Array	Any reuse services used or required by a subscribed application and its services.
xsappname	String	The xsappname configured in the security descriptor file used to create the XSUAA instance (see <a href="#">Develop the Multitenant Application [page 196]</a> ).
	String	The ID of the associated global account.
error	String	Error description for the following statuses: <ul style="list-style-type: none"> <li>• SUBSCRIBE_FAILED</li> <li>• UNSUBSCRIBE_FAILED</li> <li>• UPDATE_FAILED</li> </ul>

## Response Example

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
"subscriptions": [
    {
        "url": "http://localhost:8080/app",
        "appName": "6542fe75-2700-4401-be9a-8cb845a5dfa0",
        "consumerTenantId": "900e2f70-95f2-40d3-a32a-eaae5203cfcc6",
        "state": "SUBSCRIBED",
        "dependencies": [
            {
                "xsappname": "7d93e66e-9035-4239-a597-a739ade3473b!b220|sample-saas-reuse-service-test!b220",
                "appName": "6c13edd9-4c8a-4fc2-9e98-90445fe7a927",
                "dependencies": [
                    {
                        "xsappname": "7cab59cf-9184-4e69-a846-d930363f1f4a!b220|sample-saas-reuse-service-b-test!b220",
                        "appName": "2d46a1ac-ac15-4a8e-ab28-69608cd8fe05",
                        "dependencies": []
                    }
                ]
            }
        ]
    }
]
}

```

## 3.1.10.6.6 Update Subscription Dependencies

The API is used to update the dependencies of a multitenant application.

This REST call is asynchronous.

After executing the API, a job to update dependencies is created.

You can update dependencies for tenants with status:

- SUBSCRIBED
- UPDATE\_FAILED

### Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

### Request

**URI:** <"saas-registry"."credentials"."saas\_registry\_url">/saas-manager/v1/application/tenants/<tenantId>/subscriptions

#### URI Path Parameter

Parameter Name	Parameter Type	Description
tenantId	String	The ID of the tenant whose subscription dependencies you are updating.

**HTTP Method:** *PATCH*

#### Request Headers

Header	Required	Value
Authorization	Yes	Bearer <application JWT>
Content-Type	Yes	<application/json>

#### Request Parameters

None

#### Request Example

```
PATCH saas-manager/v1/application/tenants/900e2f70-95f2-40d3-a32a-eaae5203cfcc6/
subscriptions HTTP/1.1
Authorization: eyJhbGciOiJSUzI1NiIs...
Content-Type: application/json
```

## Response

Content Type: *TEXT*

### Response Headers

Header	Values
Content-Type	<text/plain; charset=UTF-8>
Location	<path to job status>

#### i Note

The <path to job status> serves as the relative path for the base request URI of the [Get Subscription Job Information \[page 223\]](#) API.

Use the Get Subscription Job Information API to query the status of the update you created by calling the current API.

### Response Status and Error Codes

Code	Description
202	Process to update subscriptions started.
400	Tenant not found. Reasons: <ul style="list-style-type: none"><li>• Invalid request body</li><li>• Subscription exists in status:<ul style="list-style-type: none"><li>◦ IN_PROCESS</li><li>◦ SUBSCRIBE_FAILED</li><li>◦ UNSUBSCRIBE_FAILED</li></ul></li></ul>
401	Invalid JWT.
500	Internal server error.

**Response Payload Content:** Job for update subscription of application: <*appId*> and tenant: <*tenantId*>, was created.

Where:

- <*appId*> is the ID of the application to which you subscribed the tenant.
- <*tenantId*> is the ID of the subscribed tenant.

### Response Example

```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=UTF-8
Location: /api/v2.0/jobs/afd10d2d-f235-4625-817d-cc2acac90de4
Job for update subscription of application: 6542fe75-2700-4401-be9a-8cb845a5dfa0
and tenant: 900e2f70-95f2-40d3-a32a-eaae5203cf6, was created
```

## 3.1.10.6.7 Get Subscription Job Information

Describes how to get information about the job created after one of the following APIs has been called:

- [Subscribe the Tenant to an Application \[page 213\]](#)
- [Unsubscribe the Tenant from an Application \[page 216\]](#)
- [Update Subscription Dependencies \[page 221\]](#)

### Process

The `<URL to get job status>` you received in the response headers of one of the three APIs serves as the relative path for the base request URI of this API.

### Prerequisites

You have obtained the access token for the application. See [Get an Application Access Token \[page 209\]](#).

### Request

**URI:** `<"saas-registry"."credentials"."saas_registry_url">/api/v2.0/jobs/<jobId>`

Where: `/api/v2.0/jobs/<jobId>` is the `<URL to get job status>`.

**HTTP Method:** `GET`

#### Request Headers

Header	Required	Value
Authorization	Yes	Bearer <code>&lt;application JWT&gt;</code>

#### Request Parameters

None

#### Request Example

```
GET saas-manager/v1/application HTTP/1.1
Authorization: eyJhbGciOiJSUzI1NiIs...
```

### Response

**Content Type:** `JSON`

## Response Headers

Header	Values
Content-Type	<application/json; charset=UTF-8>

## Response Status and Error Codes

Code	Reason
200	Successfully returned information about the job.
401	Invalid JWT.
500	Internal server error.

## Response Properties

Property Name	Property Type	Description
id	String	The ID of the job.
state	String	The state of the job. Values: <ul style="list-style-type: none"><li>• CREATED</li><li>• STARTED</li><li>• SUCCEEDED</li><li>• FAILED</li></ul>
createdBy	String	The service instance ID of the SaaS Provisioning ( <code>saas-registry</code> ) service that the application is using.
error	String	Error message when the state is FAILED.

## Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
{
  "id": "e1358f34-427a-4d64-94bc-15362fac528",
  "state": "SUCCEEDED",
  "createdBy": "7c1e5e62-2cba-40f1-a58a-338ce2903fc5"
}
```

## 3.1.11 Developing Node.js in the Cloud Foundry Environment

This section offers selected information for Node.js development on the SAP Cloud Platform Cloud Foundry environment and references to more detailed sources.

In this section about Node.js development, you get information about the buildpack supported by SAP and about the Node.js packages, and how you consume them in your application.

There is also a small tutorial with an introduction to securing your application, and some tips and tricks for developing and running Node.js applications on the Cloud Foundry environment.

## Node.js Buildpack

SAP Cloud Platform uses the standard Node.js buildpack provided by Cloud Foundry 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. As the Node.js packages used for development in the Cloud Foundry are similar or the same, as those in SAP HANA XS advanced, the links guide you to the SAP HANA Developer Guide for XS Advanced Model.

### Additional information:

- [The SAP NPM Registry](#)
- [Standard Node.js Packages](#)
- [Download and Consume SAP Node.js Packages](#)

## Node.js Tutorial

The tutorial will guide you through creating a Node.js application, and setting up authentication and authorization checks. This is by no means a setup for productive use, but you get to know the basics and links to some further reading.

[Create a Node.js Application \[page 226\]](#)

[Authentication Checks in Node.js Applications \[page 228\]](#)

[Authorization Checks in Node.js Applications \[page 232\]](#)

## Tips and Tricks

For selected tips and tricks for your Node.js development, see [Tips and Tricks for Node.js Applications \[page 235\]](#).

### 3.1.11.1 Create a Node.js Application

This tutorial guides you through creating and setting up a sample Node.js application by using the Cloud Foundry command line interface (cf CLI).

#### Prerequisites

- A registered user and space in the Cloud Foundry environment.
- cf CLI is installed locally, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- Node.js  is installed and configured locally, see [npm documentation](#) .
- The SAP npm registry is configured, see [The SAP NPM Registry](#).

#### Context

We start by building and deploying a simple web application that returns some sample data.

#### Procedure

1. Create a new directory named `node-tutorial`.
2. Create a `manifest.yml` file in the `node-tutorial` directory with the following content:

```
---  
applications:  
- name: myapp  
  routes:  
  - route: <host>.cfapps.<region>.hana.ondemand.com  
    path: myapp  
    memory: 128M
```

Exchange the `<host>` with a unique name, so it does not clash with other deployed application. Replace `<region>` with the location of your subaccount. For example, `eu10` designates the Europe (Frankfurt) region, see [Regions and API Endpoints Available for the Cloud Foundry Environment](#)

This configuration is used to describe the applications and how they will be deployed.

#### → Tip

For information about the `manifest.yml` file, see [Deploying with Application Manifests](#) .

3. Create a new directory inside `node-tutorial` named `myapp` and change the current directory to `myapp` with `cd myapp`.
4. Execute `npm init`.

This will walk you through creating a `package.json` file in the `myapp` directory.

- Run `npm install express --save`.

This will add the express package as a dependency in the `package.json` file. After the installation is complete, the content of the `package.json` should look similar to this:

```
{  
  "name": "myapp",  
  "version": "1.0.0",  
  "description": "My App",  
  "main": "index.js",  
  "scripts": {  
    "test": "<test>"  
  },  
  "author": "",  
  "license": "ISC",  
  "dependencies": {  
    "express": "^4.15.3"  
  }  
}
```

- Add engines and update scripts sections to the `package.json` so it looks similar to this:

```
{  
  "name": "myapp",  
  "description": "My App",  
  "version": "0.0.1",  
  "private": true,  
  "dependencies": {  
    "express": "^4.15.3"  
  },  
  "engines": {  
    "node": "10.x.x"  
  },  
  "scripts": {  
    "start": "node start.js"  
  }  
}
```

- Inside the `myapp` directory create another file called `start.js` with the following content:

```
const express = require('express');  
const app = express();  
app.get('/', function (req, res) {  
  res.send('Hello World!');  
});  
const port = process.env.PORT || 3000;  
app.listen(port, function () {  
  console.log('myapp listening on port ' + port);  
});
```

This creates a very simple web application returning “Hello World” as a response. Express is one of the most widely used Node.js modules (for serving web content) and it is the web server part of this application. After these steps are complete, note that the package `express` has been installed in the `node_modules` directory.

- Deploy the application on Cloud Foundry. Execute the following command in the `node-tutorial` directory:

```
cf push
```

### i Note

`cf push` is always executed in the same directory where the `manifest.yml` is located.

Running on the China (Shanghai) region: it is mandatory for Cloud Foundry applications deployed in SAP to be self-contained – they need to carry all of their dependencies so that the staging process does not require any network calls.

Running on the AWS, Azure, or GCP regions: the overall recommendation for Cloud Foundry applications deployed in SAP is for them to be self-contained – they need to carry all of their dependencies so that the staging process does not require any network calls.

See [Tips and Tricks for Node.js Applications \[page 235\]](#).

When the staging and deployment steps are complete you can check the state and URL of your application by using the cf app command.

```
> cf app myapp
Getting apps in org "orgname" / space "myspace" as "user"...
name:          myapp
requested state:  started
instances:      1/1
usage:          1G x 1 instances
routes:         <myapp>.cfapps.sap.hana.ondemand.com
last uploaded:  Mon 15 May 16:25:57 EEST 2017
stack:          cflinuxfs2
buildpack:      node.js 1.5.32
    state      since            cpu      memory      disk
details
#0  running    <date and time>    0.2%   57.4M of 1G  118.5M of 1G
```

9. Open a browser window and enter your application's URL.

You should see the message “Hello World!”.

## Results

The sample app is now deployed and running on Cloud Foundry.

### 3.1.11.2 Authentication Checks in Node.js Applications

#### Prerequisites

You completed the [Create a Node.js Application \[page 226\]](#) tutorial and deployed the sample application on the Cloud Foundry environment.

#### Context

Authentication in the Cloud Foundry environment is provided by the Authorization and Trust Management (XSUAA) service. In this example, OAuth 2.0 is used as the authentication mechanism. The simplest way to add authentication is to use the `@sap/approuter` package, which is a component used to provide a central entry

point for business applications. More details on the security in SAP Cloud Platform can be found in [Web Access in the Cloud Foundry Environment \[page 1104\]](#) documentation.

## Procedure

1. Create the file `xs-security.json` (see the related link) in the `node-tutorial` directory with the following content:

```
{  
  "xsappname": "myapp",  
  "tenant-mode": "dedicated"  
}
```

2. Create a `xsuaa` service instance with plan `application` which is named `myuaa`:

```
cf create-service xsuaa application myuaa -c xs-security.json
```

3. Add the `myuaa` service in `manifest.yml` file so it looks similar to this:

```
---  
applications:  
- name: myapp  
  routes:  
    - route: <host>.cfapps.<region>.hana.ondemand.com  
  path: myapp  
  memory: 128M  
  services:  
    - myuaa
```

In this case, a `myuaa` service instance will be bound to the `myapp` application during deployment.

4. Create a directory named `web` in the `node-tutorial` directory.
5. Inside the `web` directory, create a subdirectory named `resources`. This directory will be used to provide the business application static resources.
6. Inside `resources`, create an `index.html` file with the following content:

```
<html>  
<head>  
  <title>JavaScript Tutorial</title>  
</head>  
<body>  
  <h1>JavaScript Tutorial</h1>  
  <a href="/myapp/">myapp</a>  
</body>  
</html>
```

This will be the application start page.

7. Create a `package.json` file in the `web` directory by executing `npm init`.
8. Execute `npm install @sap/approuter --save` to install the `approuter` package in `web/node_modules/@sap`.
9. Replace the `scripts` section of the `package.json` file in the `web` directory with the following:

```
"scripts": {  
  "start": "node node_modules/@sap/approuter/approuter.js"  
}
```

10. Add the web application to your project.

Insert the following content at the end of your `node-tutorial/manifest.yml` file.

```
- name: web
  routes:
  - route: <host>.cfapps.<region>.hana.ondemand.com
    path: web
    memory: 128M
    env:
      destinations: >
        [
          {
            "name": "myapp",
            "url": "<myapp url>",
            "forwardAuthToken": true
          }
        ]
    services:
    - myuaa
```

- Exchange the `<host>` with a unique name, so it does not clash with other deployed applications.  
Replace `<region>` with the location of your subaccount.
- The `destinations` environment variable defines the destinations of the microservices the application router will forward requests to. Set the `url` property to the URL of the `myapp` application as displayed by the `cf apps` command, and add the network protocol (`https://`).
- In the services section we specify the `xsuaa` service name that will be bound to the application.

11. Create the `xs-app.json` file in the `web` directory with the following content:

```
{
  "routes": [
    {
      "source": "^/myapp/(.*)$",
      "target": "$1",
      "destination": "myapp"
    }
  ]
}
```

**i Note**

With this configuration, the incoming request is forwarded to your application `myapp`, configured as destination. By default, every route requires OAuth authentication, so the requests to this path will require an authenticated user.

12. Execute the following commands in the `myapp` directory to download the `@sap/xssec`, `@sap/xsenv`, and `passport` packages:

```
npm install @sap/xssec --save
npm install @sap/xsenv --save
npm install passport --save
```

13. Verify that the request is authenticated. Check the JWT token in the request using the `JWTStrategy` provided by the `@sap/xssec` package. To do that, replace the content of the `myapp/start.js` file with the following:

```
const express = require('express');
const passport = require('passport');
```

```

const xsenv = require('@sap/xsenv');
const JWTStrategy = require('@sap/xssec').JWTStrategy;
const app = express();
const services = xsenv.getServices({ uaa:'myuaa' });
passport.use(new JWTStrategy(services.uaa));
app.use(passport.initialize());
app.use(passport.authenticate('JWT', { session: false }));
app.get('/', function (req, res, next) {
  res.send('Application user: ' + req.user.id);
});
const port = process.env.PORT || 3000;
app.listen(port, function () {
  console.log('myapp listening on port ' + port);
});

```

14. Go to the node-tutorial directory and execute the following command:

```
cf push
```

This command will update the myapp application and deploy the web application.

#### i Note

From this point in the tutorial, the URL of the web application will be requested instead of the myapp URL. It will then forward the requests to the myapp application.

15. Use the cf apps command to find the URL of the web application. Open it in a Web browser.

16. Enter the credentials of your SAP Cloud Platform user.

17. Click the myapp link.

You should see the user that you logged in with in the browser window:

```
Application user: <login user>
```

18. Check that the myapp application is not accessible without authentication. Open the URL of the myapp application in a Web browser and you should get error 401 Unauthorized.

#### i Note

Both the myapp and web applications are bound to the same Authorization and Trust Management (XSUAA) service instance myuaa. In this scenario, authentication is handled by XSUAA through the application router.

## Related Information

[The Application Security Descriptor \[page 253\]](#)

[Monitoring and Troubleshooting \[page 1130\]](#)

### 3.1.11.3 Authorization Checks in Node.js Applications

#### Prerequisites

You completed the [Authentication Checks in Node.js Applications \[page 228\]](#) tutorial and deployed the sample application on the Cloud Foundry environment.

#### Context

Authorization in the Cloud Foundry environment is provided by the Authorization and Trust Management (XSUAA) service. In the previous tutorial, the `@sap/approuter` package was added to provide a central entry point for the business application and enable authentication.

In this tutorial, we extend the sample by adding authorization through the implementation of the `users` REST service. Different authorization checks will be introduced for the GET and CREATE operations to demonstrate how authorization works. The authorization concept includes elements such as roles, scopes, and attributes provided in the security descriptor file `xs-security.json`, explained in detail in the [What Is Authorization and Trust Management? \[page 1096\]](#) section.

##### i Note

Authorization checks can be configured in the application router. Check the [Routing Configuration File \[page 86\]](#), `route`'s `scope` property. This tutorial focuses on authorization checks in the microservices using the container security API for Node.js.

#### Procedure

1. To introduce application roles, open the `xs-security.json` in the `node-tutorial` directory, and add scopes and role templates as follows:

```
{  
  "xsappname": "myapp",  
  "tenant-mode": "dedicated",  
  "scopes": [  
    {  
      "name": "$XSAPPNAME.Display",  
      "description": "Display Users"  
    },  
    {  
      "name": "$XSAPPNAME.Update",  
      "description": "Update Users"  
    }  
  ],  
  "role-templates": [  
    {  
      "name": "Viewer",  
      "description": "View Users",  
      "scope-references": [  
        "$XSAPPNAME.Display"  
      ]  
    }  
  ]  
}
```

```

},
{
  "name": "Manager",
  "description": "Maintain Users",
  "scope-references": [
    "$XSAPPNAME.Display",
    "$XSAPPNAME.Update"
  ]
}
]
}

```

Two roles are introduced: `Viewer` and `Manager`. These roles are sets of OAuth 2.0 scopes or actions. The scopes are used later in the microservices code for authorization checks.

2. Update the XSUAA service.

```
cf update-service myuaa -c xs-security.json
```

3. Add a new file called `users.json` to the `myapp` directory with the following content:

```
[
  {
    "id": 0,
    "name": "John"
  },
  {
    "id": 1,
    "name": "Paul"
  }
]
```

This will be the initial list of `users` for the REST service.

4. Add a dependency to `body-parser` that will be used for JSON parsing. In the `myapp` directory execute:

```
npm install body-parser --save
```

5. Change the `myapp/start.js` by adding GET and POST operations for the `users` REST endpoint:

```

const express = require('express');
const passport = require('passport');
const bodyParser = require('body-parser');
const xsenv = require('@sap/xsenv');
const JWTStrategy = require('@sap/xssec').JWTStrategy;
const users = require('./users.json');
const app = express();
const services = xsenv.getServices({ uaa: 'myuaa' });
passport.use(new JWTStrategy(services.uaa));
app.use(bodyParser.json());
app.use(passport.initialize());
app.use(passport.authenticate('JWT', { session: false }));
app.get('/users', function (req, res) {
  var isAuthorized = req.authInfo.checkScope('$XSAPPNAME.Display');
  if (isAuthorized) {
    res.status(200).json(users);
  } else {
    res.status(403).send('Forbidden');
  }
});
app.post('/users', function (req, res) {
  const isAuthorized = req.authInfo.checkScope('$XSAPPNAME.Update');
  if (!isAuthorized) {
    res.status(403).json('Forbidden');
    return;
  }
  var newUser = req.body;
  // Process the new user and save it to the database
  // ...
  res.status(201).json(newUser);
});

```

```

    newUser.id = users.length;
    users.push(newUser);
    res.status(201).json(newUser);
  });
const port = process.env.PORT || 3000;
app.listen(port, function () {
  console.log('myapp listening on port ' + port);
});

```

Authorization checks are enforced by the `@sap/xssec` package.

To every request object, using `passport` and `xsssec.JWTSecurity`, a security context is attached as an `authInfo` object. The resulting request object is initialized with the incoming JWT token. For a full list of methods and properties of the security context, see [Authentication for Node.js Applications \[page 118\]](#)

As defined in `myapp/start.js`, for HTTP GET requests users need the scope `Display` to be authorized.

For the HTTP POST requests, the user needs to have the `Update` scope assigned.

6. Update the UI to be able to send POST requests. Change the content of `web/resources/index.html` with the following code:

```

<html>
<head>
  <title>JavaScript Tutorial</title>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/
jquery.min.js"></script>
  <script>
    function fetchCsrfToken(callback) {
      jQuery.ajax({
        url: '/myapp/users',
        type: 'HEAD',
        headers: { 'x-csrf-token': 'fetch' }
      })
      .done(function(message, text, jqXHR) {
        callback(jqXHR.getResponseHeader('x-csrf-token'))
      })
      .fail(function(jqXHR, textStatus, errorThrown) {
        alert('Error fetching CSRF token: ' + jqXHR.status + ' ' +
errorThrown);
      });
    }

    function addNewUser(token) {
      var name = jQuery('#name').val() || '--';
      jQuery.ajax({
        url: '/myapp/users',
        type: 'POST',
        headers: { 'x-csrf-token': token },
        contentType: 'application/json',
        data: JSON.stringify({ name: name })
      })
      .done(function() {
        alert( 'success' );
        window.location = '/myapp/users'
      })
      .fail(function(jqXHR, textStatus, errorThrown) {
        alert('Error adding new user: ' + jqXHR.status + ' ' + errorThrown);
      });
    }

    function addUser() {
      fetchCsrfToken(addNewUser);
    }
  </script>
</head>
<body>
  <h1>JavaScript Tutorial</h1>

```

```
<a href="/myapp/users">Show users</a>
<br/>
<br/>
<input type="text" id="name" placeholder="Type user name"></input>
<input type="button" value="Add User" onClick="javascript: addUser()"></input>
</body>
</html>
```

The UI contains a link to get all users, an input box to enter a user name, and a button to send create a new user requests.

In the sample, the code seems more complicated than expected. Clicking the *Add User* button tries to fetch a CSRF token and on success sends a POST request with the users' data as a JSON body. As we are using the application router you need to get a CSRF token before sending the POST request. This token is required for all requests that change the state.

7. Go to the `node-tutorial` directory and deploy both applications. Execute the following command:

```
cf push
```

8. Find the URL of the application router web application by using the `cf apps` command and open it in a Web browser.
9. Enter the credentials of your SAP Cloud Platform user.
10. Click the *Show users* link. This should result in a `403 Forbidden` response due to missing privileges.
11. Configure the role collections and assign user groups in the SAP Cloud Platform cockpit.

#### i Note

The configuration is not part of this tutorial. See [Set Up Security Artifacts \[page 273\]](#).

Initially, add only the `Viewer` role to the user. In a new browser window open the application URL and ensure you are logged in interactively. Click on the *Show users* link in the UI and you should be able to see a JSON response from the REST service. Clicking on *Add User* leads to a message box with error text: `403 Forbidden`.

12. Configure the user with the `Manager` role. Open the application URL in a new browser window and ensure you are logged in interactively. Now typing an example name into the edit box and clicking on the *Add user* button results in a message box: `success`. Click `OK` to see the users list. It contains the example name you entered and the users defined in the `myapp/users.json`.

### 3.1.11.4 Tips and Tricks for Node.js Applications

#### Get to know the Node.js buildpack for the Cloud Foundry environment

Check the Tips and Tricks for Node.js applications in the [Cloud Foundry Node.js Buildpack](#) documentation.

## Vendor Application Dependencies

Vendoring Node.js application dependencies is discussed in the documentation for the Cloud Foundry environment. Even though the SAP Cloud Platform is a connected environment, for productive applications the recommendation it is mandatory to vendor application dependencies.

There are various reasons for this, for example for productive applications usually security scans are run, at the same time `npm` does not provide reliable dependency fetch and you might end-up with different dependencies in case they are installed during deployment. Additionally, `npm` downloads any missing dependencies from its registry. If this registry is not accessible for some reason, the deployment may fail.

### i Note

Be aware when using dependencies containing native code, that you need to preinstall it in the same environment as the Cloud Foundry container or that the package has built-in support for it.

To ensure that prepackaged dependencies are pushed to the Cloud Foundry environment and On-premise runtime, make sure that `node_modules` directory is not listed in `.cfignore` file. It's also preferable that development dependencies are not deployed for productive deployments. To ensure that run this command:

```
npm prune --production
```

## Out of Memory at runtime

For performance reasons Node.js (V8) has lazy garbage collection. Even though there are no memory leaks in your application, this might lead to occasional restarts as explained in the [Tips for Node.js Applications](#) section in the Node.js Buildpack documentation for the Cloud Foundry Environment.

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.

You can either use `<OPTIMIZE_MEMORY>` environment variable supported by Node.js buildpack or specify the V8 heap size directly in your application start command (recommended), example for application started with 256M of RAM:

### ↳ Sample Code

```
node --max_old_space_size=192 server.js
```

You can optimize V8 behavior using additional options. You can list them using the command:

```
node --v8-options
```

## Specify Application Memory in `manifest.yml`

When deploying an application in the Cloud Foundry environment without specifying the application memory requirements, the controller of the Cloud Foundry environment will assign the default (1G of RAM currently) for

your application. Many Node.js applications require less memory and assigning the default is waste of resources.

To save memory from your quota, specify the memory size in descriptor of the deployment in the Cloud Foundry environment – `manifest.yml`. Details how to do that, can be found in the [Deploying with Application Manifests](#) topic in the documentation for the Cloud Foundry environment.

### 3.1.12 Developing Python in the Cloud Foundry Environment

This section offers selected information for Python development on the SAP Cloud Platform Cloud Foundry environment and references to more detailed sources.

In this section about Python development, you get information about the buildpack supported by SAP and about the Python packages, and how you consume them in your application.

There is also a small tutorial with an introduction to securing your application, and some tips and tricks for developing and running Python applications on the Cloud Foundry environment.

#### Python Community Buildpack

SAP Cloud Platform uses the standard Python buildpack provided by the Cloud Foundry environment 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, from the SAP Service Marketplace (SMP). For more information about how to download and use Python packages from the SAP Service Marketplace, log on to the SMP and search for the 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.

Python packages

Package	Description
<code>sap_instance_manager</code>	Python package for creating and deleting service instances per tenant within an application at runtime.

Package	Description
sap_audit_logging	Provides audit logging functionalities for Python applications.
sap_xssec	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>
hdbccli	The SAP HANA Database Client, provides means for database connectivity.

## Python Tutorial

The tutorial will guide you through creating a Python application, consuming Cloud Foundry services, and setting up authentication and authorization checks. This is by no means a setup for productive use, but you get to know the basics and links to some further reading.

[Create a Python Application \[page 238\]](#)

[Consume Cloud Foundry Services \[page 241\]](#)

[Authentication Checks in Python Applications \[page 243\]](#)

[Authorization Checks in Python Applications \[page 246\]](#)

## Tips and Tricks

Selected tips and tricks for your Python development. See [Tips and Tricks for Python Applications \[page 248\]](#).

### 3.1.12.1 Create a Python Application

#### Prerequisites

Before you start you will need to fulfill the following requirements:

- A registered user and space in the Cloud Foundry environment.
- The Cloud Foundry command line interface is installed locally. See [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).

- Python version 3.5 or higher is installed locally. See the installation guides for [OS X](#), [Windows](#), and [Linux](#).
- `virtualenv` is installed locally. See <https://github.com/kennethreitz/python-guide/blob/master/docs/dev/virtualenvs.rst>.

## Context

This tutorial will guide you through creating and setting up a simple Python application by using the Cloud Foundry command line interface (cf CLI). The tutorial will showcase some basic SAP provided Python libraries aimed to ease your application development. You will start by building and deploying the web application that returns some sample data.

## Procedure

1. Log on to Cloud Foundry. See [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
2. Create a new directory named `python-tutorial`.
3. Create a `manifest.yml` file in the `python-tutorial` directory with the following content:

### ↳ Source Code

`manifest.yml`

```
---
applications:
- name: myapp
  host: <host>
  path: .
  memory: 128M
  command: python server.py
```

Exchange the `<host>` with an unique name, so it does not clash with other deployed applications. This file is the configuration describing your application and how it should be deployed to the Cloud Foundry. See [Deploying with Application Manifests](#).

4. Specify the Python runtime version your application will run on by creating a `runtime.txt` file with the following content:

### ↳ Source Code

`runtime.txt`

```
python-3.6.x
```

### i Note

The buildpack only supports the stable Python versions, which are listed in the [Python buildpack release notes](#).

5. The application will be a web server utilizing the Flask web framework. Specify Flask as an application dependency, by creating a `requirements.txt` file with the following content:

#### « Source Code

```
requirements.txt
```

```
Flask==0.12.2
```

6. Create a `server.py` file, which will contain the following application logic:

#### « Source Code

```
server.py
```

```
import os
from flask import Flask
app = Flask(__name__)
port = int(os.environ.get('PORT', 3000))
@app.route('/')
def hello():
    return "Hello World"
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=port)
```

This is a simple server, which will return a “Hello World” when requested. Flask is one of the most widely used Python web frameworks (for serving web content) and it is the web server part of this application.

7. Vendor dependencies.

Running on the China (Shanghai) region: it is mandatory to deploy Cloud Foundry applications in SAP as self-contained. Self-contained means the applications need to carry all of their dependencies so that the staging process doesn't require any network calls.

Running on the AWS, Azure, or GCP regions: the overall recommendation for Cloud Foundry applications deployed in SAP is for them to be self-contained. Self-contained means the applications need to carry all of their dependencies so that the staging process doesn't require any network calls.

The Python buildpack provides a mechanism for that – applications can vendor their dependencies by creating a vendor folder in their root directory and execute the following command to download dependencies in it:

```
pip download -d vendor -r requirements.txt --platform manylinux1_x86_64 --
only-binary=:all:
```

#### i Note

You should always make sure you are vendorizing dependencies for the correct platform, so if you are developing on anything other than Ubuntu, use the `--platform` flag. See [pip download](#).

8. Deploy the application on Cloud Foundry. Execute the `cf push` command in the `python-tutorial` directory.

#### i Note

`cf push` is always executed in the same directory, where the `manifest.yml` is located.

When the staging and deployment steps are complete you can check the state and URL of your application by using the `cf app` command.

9. Open a browser window and enter the URL of the application.

You should see the message Hello World.

### 3.1.12.2 Consume Cloud Foundry Services

#### Prerequisites

You've gone over and completed the [Create a Python Application \[page 238\]](#) part of the tutorial.

#### Context

In this tutorial, you connect and consume a Cloud Foundry service in your application.

You can view what services and plans are available for your application to consume by executing `cf marketplace`. For this tutorial, we use the SAP HANA Service.

#### Procedure

1. Create an instance of the SAP HANA service with the following command:

```
cf create-service hana hdi-shared myhana
```

This creates a service instance called `myhana`, from the service `hana`, with the plan `hdi-shared`.

2. Bind this service instance to the application.

- a. Modify the `manifest.yml` file:

##### ↳ Source Code

###### manifest.yml

```
---
applications:
- name: myapp
  host: <host>
  path: .
  memory: 128M
  command: python server.py
  services:
    - myhana
```

- b. To consume the service inside the application, you need to read the service settings and credentials from the application. To do that, use the python module `cfenv`. Add the following line to the `requirements.txt` file:

#### ↳ Source Code

```
requirements.txt
```

```
Flask==0.12.2
cfenv==0.5.3
hdbcli
```

- c. Modify the `server.py` file to include the following lines of code, which are used to read the service information from the environment:

#### ↳ Source Code

```
server.py
```

```
import os
from flask import Flask
from cfenv import AppEnv
app = Flask(__name__)
env = AppEnv()
port = int(os.environ.get('PORT', 3000))
hana_service = 'hana'      # change to 'hanatrial' if using HANA Cloud
trial
hana = env.get_service(label=hana_service)
@app.route('/')
def hello():
    return "Hello World"
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=port)
```

When you restage the application, the SAP HANA service instance is bound to the application and the application can connect to it.

3. Connect to SAP HANA using the SAP HANA database client or `hdbcli` module provided by SAP.

To get `hdbcli` and other SAP developed python modules:

- go to <https://support.sap.com/en/my-support/software-downloads.html>,
- select *Access downloads* located under *Support Packages and Patches*,
- in the search bar type *XS PYTHON*,
- select the *XS PYTHON 1* component.

From there you can download the *XS\_PYTHON* archive and extract it in a local directory, for example `sap_dependencies`.

```
pip download hdbcli==2.3.119 -d vendor --find-links ./sap_dependencies --
platform linux_x86_64 --only-binary=:all:
```

#### i Note

Always make sure you're vendorizing dependencies for the correct platform, so if you're developing on anything other than Ubuntu, use the `--platform` flag. See [pip download](#).

4. Modify the `server.py` file to execute a query with the `hdbcli` driver:

#### ↳ Source Code

```
server.py
```

```
import os
```

```

from flask import Flask
from cfenv import AppEnv
from hdbcli import dbapi
app = Flask(__name__)
env = AppEnv()
hana_service = 'hana'      # change to 'hanatrial' if using HANA Cloud trial
hana = env.get_service(label=hana_service)
port = int(os.environ.get('PORT', 3000))
@app.route('/')
def hello():
    if hana is None:
        return "Can't connect to HANA service '{}' - check service name?".format(hana_service)
    else:
        conn = dbapi.connect(address=hana.credentials['host'],
                              port=int(hana.credentials['port']),
                              user=hana.credentials['user'],
                              password=hana.credentials['password'],
                              encrypt='true',
                              sslTrustStore=hana.credentials['certificate'])

        cursor = conn.cursor()
        cursor.execute("select CURRENT_UTCTIMESTAMP from DUMMY")
        ro = cursor.fetchone()
        cursor.close()
        conn.close()

        return "Current time is: " + str(ro["CURRENT_UTCTIMESTAMP"])
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=port)

```

5. Push the application with `cf push`.
6. Open a browser window and enter the URL of the application.

The current HANA time is displayed.

### 3.1.12.3 Authentication Checks in Python Applications

#### Prerequisites

- You have gone over and completed, the [Create a Python Application \[page 238\]](#) and [Consume Cloud Foundry Services \[page 241\]](#) parts of the tutorial.
- You have npm installed locally.

#### Context

Authentication in the Cloud Foundry environment is provided by the UAA service. In this example, OAuth 2.0 is used as the authentication mechanism. The simplest way to add authentication is to use the @sap/approuter Node.js package, which is a component used to provide a central entry point for business applications. More details on the security in SAP Cloud Platform can be found in Web Access in the Cloud Foundry Environment documentation. To use @sap/approuter we'll create a separate Node.js micro-service to act as an entry point for the application.

## Procedure

1. Create an `xs-security.json` file for your application with the following content:

### ↳ Source Code

```
xs-security.json
```

```
{  
  "xsappname" : "myapp",  
  "tenant-mode" : "dedicated"  
}
```

2. Create a UAA service instance named `myuaa` via the following command:

```
cf create-service xsuaa application myuaa -c xs-security.json
```

3. Add the `myuaa` service in `manifest.yml` file:

### ↳ Source Code

```
manifest.yml
```

```
---  
applications:  
- name: myapp  
  host: <host>  
  path: .  
  memory: 128M  
  command: python server.py  
  services:  
    - myhana  
    - myuaa
```

The `myuaa` service instance will be bound to the `myapp` application during deployment.

4. To create a second microservice, which will be the application router, create a directory called `web` in the `python-tutorial` directory.
5. Inside the `web` directory, create a sub-directory named `resources` - this directory will be used to provide the business application's static resources.
6. Inside `resources`, create an `index.html` file with the following content:

### ↳ Source Code

```
index.html
```

```
<html>  
<head>  
  <title>Python Tutorial</title>  
</head>  
<body>  
  <h1>Python Tutorial</h1>  
  <a href="/myapp/">myapp</a>  
</body>  
</html>
```

7. Create a `package.json` file in the `web` directory by executing `npm init`.

8. Execute `npm install @sap/approuter --save`, to install the `approuter` package in `web/node_modules/@sap`.
9. Add the following scripts section to the `package.json` file in the `web` directory:

```
"scripts": {
  "start": "node node_modules/@sap/approuter/approuter.js"
}
```

10. Modify the `manifest.yml` file in the `python-tutorial` directory with the following content at the end of it:

#### «, Source Code

```
---
applications:
- name: myapp
  host: <host>
  path: .
  memory: 128M
  command: python server.py
  services:
    - myhana
    - myuaa
- name: web
  host: <host>
  path: web
  memory: 128M
  env:
    destinations: >
      [
        {
          "name": "myapp",
          "url": "<myapp url>",
          "forwardAuthToken": true
        }
      ]
  services:
    - myuaa
```

- Exchange the `<host>` with an unique name, so it does not clash with other deployed application.
- The `<destinations>` environment variable defines the destinations to the micro-services, the application router will forward requests to.
- Set the `url` property to the URL of the `myapp` application as displayed by the `cf apps` command, and add the network protocol before the URL.
- In the `services` section, specify the UAA service name, that will be bound to the application.

11. Create the `xs-app.json` file in the `web` directory with the following content:

#### «, Source Code

`xs-app.json`

```
{
  "routes": [
    {
      "source": "^/myapp/(.*)$",
      "target": "$1",
      "destination": "myapp"
    }
  ]
}
```

```
}
```

#### i Note

With this configuration, the incoming request path is connected with the destination where the request should be forwarded to. By default, every route requires OAuth authentication, so the requests to this path will require an authenticated user.

12. Navigate to the `node-tutorial` directory and execute `cf push`. This command will update the `myapp` application and will deploy the `web` application.

#### i Note

From this point in the tutorial the URL of the `web` application will be requested instead of the `myapp` URL. It will then forward the requests to the `myapp` application.

13. Check the URL of the `web` application via the `cf apps` command and open it in a browser window.

You should be prompted to log in and then you should see the current HANA time displayed by the Python application.

### 3.1.12.4 Authorization Checks in Python Applications

#### Prerequisites

You have completed [Authentication Checks in Python Applications \[page 243\]](#) and have the sample application deployed on the Cloud Foundry environment.

#### Context

Authorization in the Cloud Foundry environment is provided by the UAA service. In the previous example, the `@sap/approuter` package was added to provide a central entry point for the business application and enable authentication. Now to extend the sample, authorization will be added. The authorization concept includes elements such as roles, scopes, and attributes provided in the security descriptor file `xs-security.json`, explained in details in the [What Is Authorization and Trust Management? \[page 1096\]](#) section.

#### Procedure

1. Add the `xssec` security library to the `requirements.txt` file, to place restrictions on the content you serve.

## ↳ Source Code

requirements.txt

```
Flask==0.12.2
cfenv==0.5.3
hdbcli==2.3.14
xssec==1.0.0
```

2. Then vendor it inside the vendor folder by executing: `pip download -d vendor -r requirements.txt --find-links sap_dependencies` from the root of the application.
3. Modify `server.py` to use the security library:

## ↳ Source Code

server.py

```
import os
from flask import Flask
from cfenv import AppEnv
from flask import request
from flask import abort

import xssec
from hdbcli import dbapi
app = Flask(__name__)
env = AppEnv()
port = int(os.environ.get('PORT', 3000))
hana = env.get_service(label='hana')
uaa_service = env.get_service(name='myuaa').credentials
@app.route('/')
def hello():
    if 'authorization' not in request.headers:
        abort(403)
    access_token = request.headers.get('authorization')[7:]
    security_context = xssec.create_security_context(access_token,
uaa_service)
    isAuthorized = security_context.check_scope('openid')
    if not isAuthorized:
        abort(403)
    conn = dbapi.connect(address=hana.credentials['host'],
port=int(hana.credentials['port']), user=hana.credentials['user'],
password=hana.credentials['password'])
    cursor = conn.cursor()
    cursor.execute("select CURRENT_UTCTIMESTAMP from DUMMY", {})
    ro = cursor.fetchone()
    cursor.close()
    conn.close()
    return "Current time is: " + str(ro["CURRENT_UTCTIMESTAMP"])
if __name__ == '__main__':
    app.run(host='0.0.0.0', port=port)
```

4. Try to access the application directly, you should see an HTTP 403 error. If you however access the application through the application router, you should see the current HANA time, provided you have the scope 'openid' assigned to your user.

Since the OAuth 2.0 client is used, the scope `openid` is assigned to your user by default and you are able to access the application as usual.

The functional authorization scopes for applications are declared in the `xs-security.json`, see [Specify the Security Descriptor Containing the Functional Authorization Scopes for Your Application \[page 275\]](#).

### 3.1.12.5 Tips and Tricks for Python Applications

- Running on the China (Shanghai) region: it is mandatory for Cloud Foundry applications deployed in SAP, is for them to be deployed as self-contained – they need to carry all of their dependencies so that the staging process does not require any network calls.  
Running on the AWS, Azure, or GCP regions: the overall recommendation for Cloud Foundry applications deployed in SAP, is for them to be deployed as self-contained – they need to carry all of their dependencies so that the staging process does not require any network calls.  
See <https://docs.cloudfoundry.org/buildpacks/python/index.html#vendorizing>
- The cfenv package provides access to 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>.
- While developing Python applications (whether in the Cloud or not) it's a very good idea to use virtual environments. The most famous Python package providing such a functionality is virtualenv. See <https://virtualenv.pypa.io/en/stable/>
- The PEP 8 style guide for Python applications - <https://www.python.org/dev/peps/pep-0008/>, will help you improve your applications.

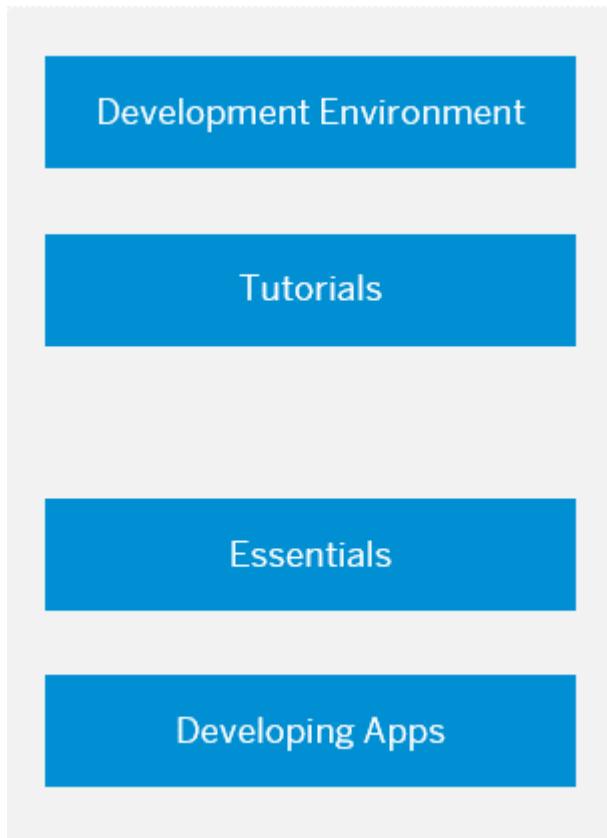
### 3.1.13 Developing SAPUI5

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\\_115/unique\\_115\\_Connect\\_42\\_subsection-im1 \[page 249\]](#)
- [#unique\\_115/unique\\_115\\_Connect\\_42\\_subsection-im2 \[page 250\]](#)
- [#unique\\_115/unique\\_115\\_Connect\\_42\\_subsection-im3 \[page 250\]](#)
- [#unique\\_115/unique\\_115\\_Connect\\_42\\_subsection-im4 \[page 250\]](#)

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 Web IDE Full-Stack

Use the SAP Web IDE Full-Stack to develop on Cloud Foundry. Even though the SAP Web IDE Full-Stack is running on the Neo Environment, you can define your space in the Cloud Foundry environment in the project settings of your project. The possibility to define space settings is limited to Multi-Target Application projects.

1. Register for an SAP Cloud Platform trial account at <https://account.hanatrial.ondemand.com/> and log on afterwards.
2. Open SAP Web IDE Full-Stack
3. Setting Up Application Projects - Create a Project from Scratch & Select a Cloud Foundry Space

## 4. Create an HTML5 Module

### Tutorials

In an HTML5 module in SAP Web IDE Full-Stack, more files are created than described in these tutorials, but you can run through it and at the end you have a running application on the Cloud Foundry environment.

For more information 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

## 3.1.14 Developing Security Artifacts

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 security descriptor file `xs-security.json`. Using the cockpit, administrators of the environment assign the authorizations to business users.

### Related Information

[SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#)

### 3.1.14.1 Maintenance of Application Security

Set up the components required in the context of application security.

You have already prepared an application that you want to deploy in the Cloud Foundry environment of SAP Cloud Platform (see the related link).

Application security is maintained in the `xs-security.json` file; the contents of the `xs-security.json` file cover the following areas:

- Scopes for functional authorization checks  
Scopes cover business users' authorizations in a specific application. For more information, see the related link.
- Attributes  
Attributes refine business users' authorizations according to attributes that come with the business users. You can use, for example fixed attribute values, such as 'country equals USA' or SAML attributes.
- Role templates  
A description of one or more roles (for example, "employee" or "manager") to apply to a user and any attributes that apply to the roles; the role template is used to build a role.
- Role collections  
A description of role collections that reference predefined role templates. Administrators can assign these role collections to users.

### Example

#### ≡, Output Code

```
AppName
| - web/
```

```
|   |- xs-app.json  
|   \- resources/  
`- xs-security.json          # Security deployment artifacts/scopes/auths  
  \- [manifest.yml]
```

## Related Information

[Development \[page 52\]](#)

[Scopes for Functional Authorization Checks \[page 252\]](#)

### 3.1.14.1.1 Authorization Concept for Users of an Application

Business users in an application of the Cloud Foundry environment at SAP Cloud Platform should have different authorizations because they work in different jobs.

For example in the framework of 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 should cover the needs of these three employee groups. This authorization concept includes elements such as roles, scopes, and attributes.

### 3.1.14.1.2 Scopes for Functional Authorization Checks

Authorization in the application router and runtime container are handled by scopes. Scopes are groupings of functional organizations defined by the application.

From a technical point of view, the resource server (the relevant security container API) provides a set of services (the resources) for functional authorizations. The functional authorizations are organized in scopes.

Scopes cover business users' authorizations in a specific application. They are deployed, for example, when you deploy an update of the application. The security descriptor file `xs-security.json` contains the application-specific “local” scopes (and, if applicable, also “foreign” scopes, which are valid for other defined applications).

Scopes provide a set of role templates for a named application. The role templates contain the authorizations for business users' activities, such as viewing, editing, or deleting data. Information retrieved from the user's identity (such as department or cost center) is stored in attributes.

After the developer has created the role templates and deployed them to the relevant application, it is the Cloud Foundry administrator's task to use the role templates to build roles, aggregate roles into role collections, and assign the role collections to business users in the application.

To assign scopes to an application, you need to perform the following steps:

1. Create an instance of the `xsuaa` service.

You can use the service broker to create an instance of the `xsuaa` service. This ensures the creation of a new OAuth 2.0 client in the UAA.

2. Bind the service instance to the application.

Scopes are configured and assigned in the application descriptor file `xs-security.json`, which can be placed in the root folder of the application or in a `security` folder in your project.

→ Tip

For access to SAP HANA objects, the privileges of the technical user (container owner) apply; for business data, instance-based authorizations, CDS access policies are used.

## Related Information

[Create a Service Instance from the xsuaa Service \[page 278\]](#)

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

### 3.1.14.1.3 The Application Security Descriptor

A file that defines the details of the authentication methods and authorization types to use for access to your application.

The `xs-security.json` file uses JSON notation to define the security options for an application; the information in the application-security file is used at application-deployment time, for example, for authentication and authorization (see the related links). Applications can perform scope checks (functional authorization checks with Boolean result) and checks on attribute values (instance-based authorization checks). The `xs-security.json` file declares the scopes and attributes on which to perform these checks. This enables the User Account and Authentication (UAA) service to limit the size of an application-specific JSON Web Token (JWT) by adding only relevant content.

Scope checks can be performed by the application router (declarative) and by the application itself (programmatic, for example, using the container API). Checks using attribute values can be performed by the application (using the container API) and on the database level.

The contents of the `xs-security.json` are used to configure the OAuth 2.0 client; the configuration is shared by all components of an SAP multi-target application. The contents of the `xs-security.json` file cover the following areas:

- Authorization scopes  
A list of limitations regarding privileges and permissions and the areas to which they apply
- Attributes  
Attributes refine business users' authorizations according to attributes that come with the business users. You can use, for example fixed attribute values, such as 'country equals USA' or SAML attributes (cloud management tools feature set A).
- Role templates  
A description of one or more roles to apply to a user and any attributes that apply to the roles

The following custom options are available:

- Tenant mode  
This option is only relevant if you use tenants.
- OAuth 2.0 configuration  
Use this property to set custom parameters for tokens.

The following example shows a simple application-security file:

### Sample Code

#### Example xs-security.json File

```
{  
    "xsappname" : "node-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"]  
        }  
    ]  
}
```

## Scopes

Scopes are assigned to users by means of security roles, which are mapped to the user group(s) to which the user belongs. Scopes are used for authorization checks by the application router.

- Application router  
URL prefix patterns can be defined and a scope associated with each pattern. If one or more matching patterns are found when a URL is requested, the application router checks whether the OAuth 2.0 access token included in the request contains at least the scopes associated with the matching patterns. If not all scopes are found, access to the requested URL is denied. Otherwise, access to the URL is granted.
- Application container  
The container security API includes the `hasScope` method that allows you to programmatically check whether the OAuth 2.0 access token used for the current request has the appropriate scope. Among other things, the access token contains a list of scopes that the user is allowed to access in the current context. Each scope name must be unique in the context of the current UAA installation.

## Attributes

You can define attributes so that you can perform checks based on a source that is not yet defined. In the example `xs-security.json` file included here, the check is based on a cost center, whose name is not known because it differs according to context.

## Role Templates

A role template describes a role and any attributes that apply to the role.

If a role template contains attributes, you must instantiate the role template. This is especially true with regards to any attributes defined in the role template and their concrete values, which are subject to customization and, as a result, cannot be provided automatically. Role templates that only contain local scopes can be instantiated without user interaction. The same is also true for foreign scopes where the scope "owner" has declared his or her consent in a kind of white list (for example, either for "public" use or for known "friends").

### i Note

The resulting application-specific role instance needs to be assigned to one or more user groups.

## Related Information

[Authentication Checks in Node.js Applications \[page 228\]](#)

### 3.1.14.1.4 Application Security Descriptor Configuration Syntax

The syntax required to set the properties and values defined in the `xs-security.json` application-security description file.

#### ↳ Sample Code

```
{
  "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",
      "$XSAPPNAME.Delete" ],
    "attribute-references" : [
      "Country",
      "CostCenter"
    ]
  }
],
  "authorities": ["$ACCEPT_GRANTED_AUTHORITIES"],
  "oauth2-configuration": {
    "token-validity": 900,
    "redirect-uris": ["http://<host_name1>","http://
<host_name2>"]
}
```

```
}
```

### → Tip

Try out the tutorials of Authorization and Trust Management to get familiar with using the application security descriptor in the Cloud Foundry environment of SAP Cloud Platform.

See [Tutorials for Authorization and Trust Management \[page 285\]](#).

## xsappname

Use the `xsappname` property to specify the name of the application that the security description applies to.

### « Sample Code

```
"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 Cloud Platform: "aA"-“zZ”, “0”-“9”, “-” (dash), “\_” (underscore), “/” (forward slash), and “\” (backslash).
- The maximum length of an application name is 128 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)	An OAuth client gets a separate client secret for each subaccount.
shared	An OAuth client always gets the same client secret. It is 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.

#### i Note

If you do not specify `tenant-mode` in the `xs-security.json`, the UAA uses the dedicated tenant mode.

## scopes

In the application security file (An OAuth client always gets the same client secret. It is valid in all subaccounts. The application service plan uses this tenant mode.`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`**local** and **foreign** scopes; that is, the permissions the application requires to be able to respond to all requests. file is used to authorize the OAuth client of the application with the correct set of

#### ↳ Sample Code

```
"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 is used to authorize the OAuth client of the application with the correct set section 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 are 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](#)An OAuth client always gets the same client secret. It is valid in all subaccounts. The. 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-asAuthorities-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-asAuthorities-to-apps" :  
[ "$XSAPPNAME(<service_plan>,<foreign_xsappname>)" ]
```

#### • Example

```
"grant-asAuthorities-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”, “-” (dash), “\_” (underscore), “/” (forward slash), “\” (backslash), “:” (colon), and the element is only relevant for user scenarios where roles are not defined in their own service instance. Example: an admin application integrates different components, each having an own “.” (dot)
- Scope names cannot start with a leading dot “.”(for example, .myScopeName1).
- The maximum length of a scope name, including the fully qualified application name, is 193 characters.

## attributes

In the application security file (`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 is only relevant for user scenarios file. You can define multiple attributes.

### Sample Code

```
"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 cockpit to assign the value of the attributes. You can use the static value.
- Attributes from an SAML identity provider  
If a SAML identity provider provides the users, you can reference a SAML assertion attribute. The SAML assertion is issued by the configured identity provider during authentication. You find the SAML attribute value in the SAML configuration of your identity provider. The attributes provided by the SAML identity provider, appear as a SAML assertion attribute in the JSON web token if the user has assigned the corresponding roles. You can use the assertion attributes to achieve instance-based authorization checks when using an SAP HANA database.
- 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.

For more information, see the related link.

The `attributes` definition can take the following parameters:

attributes Parameters

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> .  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.

## 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.

### Sample Code

```
"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.

#### ↳ Sample Code

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, cannot 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 white list (for example, either for "public" use or for known "friends").

#### i Note

The resulting (application-specific) role instances need to be assigned to the appropriate user groups.

role-template Parameters

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 The name can be up to 255 characters long. The naming conventions of role-templates do not 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, name is the name of the attribute. Use the attribute name specified in the attributes section.</p> <p>default-values, for example "default-values": ["LeaveRequestWorkflow"]. If you don't want to configure default values, do not specify the default-values element.</p> <p>For more information, see <a href="#">Relationship between default-values of attribute-references and valueRequired</a>.</p>	<p>↳ Sample Code</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 are not 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”, “\_” (underscore)
- The maximum length of a role-template name is 64 characters.

## 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 Cloud Platform.

The role-collections property only makes sense if application developers reference role templates that can create default roles at deployment time.

#### i 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 Parameters

Key	Description	Example
name	The name of the role collection to build	Employee
description	A short summary of the role collection to build	Employee roles
role-template-references	The role templates referenced by the role collections property	\$XSAPPNAME.Employee Syntax \$XSAPPNAME (<service_plan>, <xsappname>)

#### Example

\$XSAPPNAME (application, myapp2)

#### 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 is 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.

## 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.

### i 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": [  
        "http://<host_name1>",  
        "http://<host_name2>"],  
    "refresh-token-validity": 1800,  
    "system-attributes": ["groups", "rolecollections"],  
    "allowedproviders": ["origin_key1", "origin_key2"]  
}
```

The following configuration keys are available:

`oauth2-configuration` Parameters

Key	Description	Example
<code>token-validity</code>	Token validity in seconds. If you do not define a value, the default value is used.  Default: 43200 (12 hours)	900 (15 minutes)
<code>refresh-token-validity</code>	Refresh token validity in seconds. If you do not define a value, the default value is used.  Default: 2592000 (30 days)	<b>604800</b> (7 days)
<code>redirect-uris</code>	This key contains a whitelist of the allowed redirect URIs. SAP Cloud Platform performs a check of the URIs, so make sure that you enter valid URIs here.  If you use your own domain in the application route, you must specify the redirect URI. For more information, see <a href="#">Domain Checks at Browser Login/Logout [page 1106]</a> .	[ "http://<host_name1>", "http://<host_name2>" ]

Key	Description	Example
system-attributes	<p>Includes the system attributes in the JWT token. If you do not define a value, no system attributes come with the JWT token. As an option, you can include for example SAML user groups or role collections as system attributes.</p> <p>Values:</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> </ul>	<code>["groups", "rolecollections"]</code>
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  <a href="#">Security &gt; Trust Configuration</a> page of your identity providers.</p>	<code>["origin_key1", "origin_key2"]</code>

**→ Recommendation**

We recommend that you configure allowed identity providers on subaccount level. For more information, see [Provide Logon Link Help to Identity Provider for Business Users \[page 809\]](#).

Key	Description	Example
credential-types	<p>→ <b>Recommendation</b></p> <p>The xsuaa service supports multiple secrets for the same OAuth 2.0 client. They are referred as binding credentials. The secrets are structured as \$. A secret is valid as long as the binding (or service key) exists. When you delete it, it becomes invalid. We recommend that you use this approach for secret rotation.</p> <p>Since SAP ID service only supports two secrets for one OAuth 2.0 client, the secret is set when a service instance is created. In this case, all bindings use the same secret. The secret is valid as long as the instance exists. It cannot be rotated.</p>	[ "binding-secret", "instance-secret" ]

## 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>

**i Note**

Currently, you can only use the application service plan.

**❖ Example**

```
$XSAPPNAME (application, business-partner)
```

Description	Syntax	
Reference to any service instance in the same sub-account and space	<p><code>\$XSSERVICENAME (&lt;service_instance-name&gt;)</code></p> <p>This is the service instance name you used when you created it.</p>	
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" : [ "\$XSAPPNAME(application,busin ess-partner)"]</pre> <p>↳, Sample Code</p> <pre>"granted-apps" : [ "\$XSAPPNAME(application,busin ess-partner1)", "\$XSAPPNAME(application ,business-partner2)"]</pre>
grant-asAuthorities-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-asAuthorities-to- apps" : [ "\$XSAPPNAME(application,busin ess-partner)"]</pre> <p>↳, Sample Code</p> <pre>"grant-asAuthorities-to-apps" : [ "\$XSAPPNAME(application,busin ess-partner1)", "\$XSAPPNAME(application ,business-partner2)"]</pre>
authorities	Granting authorities (for a client credentials scenario)	<p>↳, Sample Code</p> <pre>"authorities": [ "\$XSAPPNAME(application,busin ess-partner)"]</pre>
scope-references	Referencing scopes in role templates	<p>↳, Sample Code</p> <pre>\$XSAPPNAME.Display</pre>

Property	Description	Example
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": ["\$XSAPPNAME(application,business-partner)"]</pre> <pre>"foreign-scope-references": ["\$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 is 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": ["&lt;attribute_value&gt;"]</pre>	<pre>"valueRequired" : true</pre> or no <code>valueRequired</code> key at all
A default role with default values is generated automatically. Administrators do not have to set attribute values for their role. A role that is not restricted by attributes (unrestricted) is created.	<pre>"default-values": ["&lt;attribute_value&gt;"]</pre>	<pre>"valueRequired" : false</pre>
A default role cannot be created automatically. Administrators must set attribute values for their roles.	No <code>default-values</code> element	<pre>"valueRequired" : true</pre> or no <code>valueRequired</code> key at all

Use Case	Role Template	Attribute
A default role is generated automatically. Administrators do not have to set attribute values for their roles. A role that is not restricted by attributes (unrestricted) is created.	No default-values element	"valueRequired" : false

## Related Information

[Attributes \[page 822\]](#)

[Granting Scope Access to Another Application \[page 282\]](#)

[Tutorials for Authorization and Trust Management \[page 285\]](#)

## 3.1.14.2 Quick Start: 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

[Developing Security Artifacts \[page 251\]](#)

[Deploy Business Applications in the Cloud Foundry Environment \[page 62\]](#)

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

[Assign Role Collections \[page 829\]](#)

[Tutorials for Authorization and Trust Management \[page 285\]](#)

### 3.1.14.3 Set Up Security Artifacts

Developers create authorization information for business users in their environment; the information is deployed in an application and made 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 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. The administrators use the role templates to build roles, which are aggregated in role collections. The role collections are 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.

#### i 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	<p>Specify the security descriptor file containing the functional authorization scopes for your application</p> <p><a href="#">Specify the Security Descriptor Containing the Functional Authorization Scopes for Your Application [page 275]</a></p> <p>(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.</p>	Application developer	Text editor
2	<p>Create role templates for the application using the security descriptor file</p> <p><a href="#">Create Role Templates for the Application [page 277]</a></p> <p><a href="#">The Application Security Descriptor [page 253]</a></p>	Application developer	CF command line interface

Step	Task	User Role	Tool
3	Create a service instance from the xsuaa service using the service broker  <a href="#">Create a Service Instance from the xsuaa Service [page 278]</a>	Application developer	CF command line interface
4	Bind the service instance to the application by including it into the manifest file  <a href="#">Bind the xsuaa Service Instance to the Application [page 280]</a>	Application developer	Text editor
5	Deploy the application  <a href="#">Deploy Business Applications in the Cloud Foundry Environment [page 62]</a>	Application developer	CF command line interface or SAP Cloud Platform cockpit

#### Setting Up Authorization Artifacts (Administrators)

Step	Task	User Role	Tool
1	Use an existing role or create a new one using role templates  <a href="#">Add Roles to Role Collections on the Application Level [page 826]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
2	Create a role collection and assign roles to it  <a href="#">Maintain Role Collections [page 828]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
3	Assign the role collections to users  <a href="#">Managing Users and Their Authorizations Using the sapcp CLI [page 873] or Assign Role Collections [page 829]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform

Step	Task	User Role	Tool
4	(If you do not use SAP ID Service) Assign the role collections to SAML 2.0 user groups (cloud management tools feature set A regions)	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit
	<a href="#">Map Role Collections to User Groups [page 831]</a>		
5	Assign the role collection to the business users provided by an SAML 2.0 identity provider (cloud management tools feature set A)	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit
	<a href="#">Directly Assign Role Collections to Users [page 830]</a>		

## Related Information

[Building Roles and Role Collections for Applications \[page 819\]](#)

[Assign Role Collections \[page 829\]](#)

[Map Role Collections to User Groups \[page 831\]](#)

### 3.1.14.3.1 Specify the Security Descriptor Containing the Functional Authorization Scopes for Your Application

The functional authorization scopes for your application need to be known to the User Account and Authentication (UAA) service. To do this, you declare the scopes of your application in the security descriptor file named `xs-security.json` and then you assign the security descriptor file to the application.

## Prerequisites

- You have the `Space Developer` role in this subaccount (see the related link).

## Context

You have created an OAuth 2.0 client using the service broker. The OAuth 2.0 client includes the following configuration:

- Supported OAuth 2.0 flows
  - Authorization code

- SAML 2.0 bearer assertion
- Client credentials
- Resource owner password credentials

#### **! Restriction**

This is only available if there are PaaS tenants and if the default SAML 2.0 identity provider, which is SAP ID service, is used.

- Authorities for client credential flow
- Scopes assigned to the client
  - openid
  - The scopes defined by the application
  - The scopes from other applications which have been granted to this OAuth 2.0 client

When you use the `cf create-service` command, you create an OAuth 2.0 client with a service instance and you specify the `xs-security.json` file that is relevant for your application. Using this security descriptor file, the OAuth 2.0 client that has been created for your application receives permission to access the scopes of your application automatically.

## **Procedure**

1. Go to the folder where you want to store your application security descriptor file.
2. Create file `xs-security.json`. We recommend that you use the root folder of the application or a subfolder called `security`. You find a description and the syntax of the application security descriptor file in the related links.

You can now create a service instance using the `xs-security.json` file.

## **Related Information**

[The Application Security Descriptor \[page 253\]](#)

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

### **3.1.14.3.2 Create Role Templates for the Application**

Using the application security descriptor file (`xs-security.json`), you define role templates, authorization scopes, and attributes.

#### **Context**

#### **Procedure**

1. Open the `xs-security.json` file.
2. Define the necessary role templates, authorization scopes, and attributes. For details , see the related links.

#### **Related Information**

[The Application Security Descriptor \[page 253\]](#)

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

### **3.1.14.3.3 Create Multiple Tenants for an Application**

To run an application for multiple tenants, you need to create copies of the OAuth 2.0 client in a identity zone.

#### **Context**

The User Account and Authentication service supports the concept of identity zones to isolate OAuth 2.0 clients, SAML trust relationships, and local users. An identity zone is identified by a unique host name and a domain. The design time artifacts like scope, attribute, role template are shared, while the run time entities role, role collection are distinct for every identity zone.

To run the same application for multiple tenants, proceed as follows:

## Procedure

1. Create a service instance in the default identity zone uaa.

```
cf create-service xsuaa default -c <path>xs-security.json
```

2. Create a copy of the OAuth 2.0 client in an identity zone.

```
cf create-service xsuaa default <instance_name> -c  
'{"xsappname":"<application_name>","identityzone":"<identity_zone_name>"}'  
cf create-service xsuaa default authorizationtest-uaa -c '{"xsappname":"node-  
uaa","identityzone":"HOST.example.com"}'
```

### 3.1.14.3.4 Create a Service Instance from the xsuaa Service

Create a service instance from the xsuaa service using the service broker.

## Context

If you want to grant users access to an application, you must create a service instance (acting as OAuth 2.0 client) for this application.

## Procedure

1. Go to the folder where the xs-security.json file is stored.
2. Create a service instance based on the xsuaa service and the service plan using the security descriptors in the xs-security.json file.

```
cf create-service <service_name> <service_plan> <service_instance_name> -c xs-  
security.json
```

#### ❖ Example

```
cf create-service xsuaa application authorizationtest-uaa -c xs-security.json
```

#### i Note

If you want to update an already existing service instance, see the related link.

## Related Information

[Update a Service Instance \[page 279\]](#)

[The Application Security Descriptor \[page 253\]](#)

### 3.1.14.3.4.1 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.

#### 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
```

### 3.1.14.3.4.2 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>
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>

#### i Note

Do not change the valueType of the attribute.

### 3.1.14.3.5 Bind the xsuaa Service Instance to the Application

You must bind the service instance created by you to your application using the manifest file.

#### Context

You must bind the service instance that acts as OAuth 2.0 client to your application. You find it under `name` in the related manifest file. This file is located in the folder of your application, for example `hello-world`.

## Procedure

1. If you haven't already done so, deploy the application. Go to the root folder of your application and use the following command:

```
cf push <APP>
```

2. To bind the service instance to your application, use the following command:

```
cf bind-service <APP> <SERVICE_INSTANCE>
```

### ❖ Example

```
cf bind-service hello-world xsuaa
```

You have now deployed your application with the authorization artifacts that have been created earlier. The application also has an OAuth 2.0 client.

### 3.1.14.3.5.1 Retrieve Credentials for Remote Applications

You need to get the credentials of a service instance for an application that runs outside of the instance of the Cloud Foundry environment at SAP Cloud Platform.

#### Prerequisites

- A service instance for your application is available.

#### Context

##### i Note

Applications that run inside the instance of the Cloud Foundry environment at SAP Cloud Platform get their credentials after the respective service has been bound to the application. However, you cannot use binding for an application that runs outside of the instance of the Cloud Foundry environment at SAP Cloud Platform.

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.

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, 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).

## Procedure

1. Create a service key for the service instance of the remote application.

```
cf create-service-key <service_instance_name> <service-key-name>
```

### ❖ Example

```
cf create-service-key rem-app-service rem-app-sk
```

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 Code

```
{
  "clientid" : "sb-sample-app!i1",
  "verificationkey" : "-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIIBCgKCAQEAYi7TbPYgsIV5t2WdkabI/
XWoYrTTzIZCj0mxbp+LMdVCrnufbT9bwZ2ZJ1HB3x0DKk3Os9g2CFiB/2yCMm/
jJe2CJJ06zhYGRIZwSu6r7Is7R4TEs8bhCQEY25LvEbK2qvZMUjxcju
+13Vkn9NYViF9PRhbMxRX7i9OJeBHn/
JyYFvvBxUnCiFiLpnMiNB0tDkNf0EcPd3TWvmR8KwQGNnPT5ccpMD5fKQoC/K5wVy
+bWa43Z1d3AA4QYBPVQX+PcWifulf7xtZVqLPMDE4Q8eWQYaiGkk6A+RO0RCIJ/
byMbvm50SPe8S6+obB/3j0eJ4b7phGFjpZbTv1UQIDAQAB-----END PUBLIC KEY-----",
  "xsappname" : "sample-app!i1",
  "identityzone" : "uaa",
  "clientsecret" :
  "+Pf4sw356jkfDx1k9Hz2nX1gH3sysb2sOW627XOjwUyc2F550YagpiaPrk
+AdbqBwTSi8a5qWLPI\njQvQpTPlia==",
  "url" : "https://host.acme.com:30632/uaa-security"
}
```

## 3.1.14.3.6 Granting Scope Access to Another Application

Another application wants to use the scope of an existing application. For this purpose, the existing application needs to grant access to the scope for the application that wants to use this scope.

Let's assume that your application name is `sample-leave-request-app`. This application provides a function for creating leave requests (for all employees), for approving leave requests (only for managers), and scheduling some jobs. The `MobileApprovals` application wants to use the scopes of the `sample-leave-request-app`. Let's also assume that the exact scope names are not yet defined. In this case, the `xs-security.json` file of your application might resemble the following code example.

## i Note

The `xs-security.json` file only includes own scopes (starting with `$XSAPPNAME.`) in the `scopes` section. Scopes provided by other applications (like the `JobScheduler` scope) are only referenced in the `role-template`.

## ↳ Sample Code

Example `xs-security.json` file

```
{  
  "xsappname"      : "sample-leave-request-app",  
  "scopes"        : [ { "name"           : "$XSAPPNAME.createLR",  
                      "description"    : "create leave requests" },  
                     { "name"           : "$XSAPPNAME.approveLR",  
                      "description"    : "approve leave requests",  
                      "granted-apps"   : "  
[ \"$XSAPPNAME(application,mobile-subaccount-id,MobileApprovals)\" ]  
                      } ],  
  "attributes"     : [ { "name"           : "costcenter",  
                      "description"    : "costcenter",  
                      "valueType"       : "s"  
                     } ],  
  "role-templates": [ { "name"           : "employee",  
                      "description"    : "Role for creating leave  
requests",  
                      "scope-references": [  
                        "$XSAPPNAME.createLR", "JobScheduler.scheduleJobs" ],  
                        "attribute-references": [ "costcenter" ] },  
                     { "name"           : "manager",  
                      "description"    : "Role for creating and  
approving leave requests",  
                      "scope-references": [  
                        "$XSAPPNAME.createLR", "$XSAPPNAME.approveLR", "JobScheduler.scheduleJobs" ],  
                        "attribute-references": [ "costcenter" ] }  
                    ]  
}
```

In this example, the `sample-leave-request-app` application grants another application (`MobileApprovals` in the `mobile-subaccount-id`) access to one of its scopes (`sample-leave-request-app.approveLR`).

The `role-template` section in the `xs-security.json` of the `MobileApprovals` application contains the reference to the scope granted by the `sample-leave-request-app` application.

## ↳ Sample Code

Example `xs-security.json` file

```
"scope-references" : [ "$XSAPPNAME(application,subaccount-id,sample-leave-  
request-app).approveLR" ]
```

### 3.1.14.3.6.1 Valid Redirect URIs for OAuth 2.0 Authorization Code Flow

After logging on to an application, you want to be redirected exactly to the page of the application in question. It should therefore not be possible to use an open redirect, which might take you to the wrong page for example, or to a malicious page.

At runtime, the User Account and Authentication service checks the redirect URI for correctness and rejects access attempts to incompatible redirect URIs. This is possible because the security descriptor file `xs-security.json` contains the correct redirect URI.

#### ↳ Sample Code

```
"oauth2-configuration": {  
    "redirect-uris": ["http://<host_name1>","http://<host_name2>"]  
}
```

For more information, see the related links.

The User Account and Authentication service stores the correct redirect URIs in the OAuth client table. To avoid this kind of arbitrary redirect after a logon request, the UAA checks the redirect URI and thus makes sure that users access the correct page.

### Related Information

[Configure Redirect URLs for Browser Logout \[page 1108\]](#)

[Browser Redirects Using Wildcards \[page 1110\]](#)

### 3.1.14.4 Tutorials for Authorization and Trust Management

Follow the tutorials below to get familiar with Authorization and Trust Management in the Cloud Foundry environment of SAP Cloud Platform.

#### Tutorials for Authorization and Trust

#### Management in the Cloud Foundry

#### environment

#### Language / Framework

#### Link

Learn how to secure a basic single-tenant Node.js application with the Authorization and Trust Management service. 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 with the Authorization and Trust Management service. 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 Cloud Platform with the Authorization and Trust Management service (XSUAA) 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>
Learn how to secure microservices in SAP Cloud Platform with the Authorization and Trust Management service (XSUAA). This sample provides J2EE Configuration using web.xml and uses the SAP Java Buildpack.	J2EE, SAP Java Buildpack	<a href="#">GitHub</a>

## Tutorials for Authorization and Trust

### Management in the Cloud Foundry

environment	Language / Framework	Link
Learn how to use java-security 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.	Java	<a href="#">GitHub</a>
Learn in this reference application how the Authorization and Trust Management service fits into a complete architecture of microservices that interact with each other.	Java	<a href="#">GitHub</a>

### 3.1.15 Multitarget Applications in the Cloud Foundry Environment

A Multitarget application (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. Afterward, the MTA deployment service validates, orchestrates, and automates the deployment of the MTA, which results in Cloud Foundry applications, services and SAP specific contents. For more information about the Multitarget Application model, see the official [The Multitarget Application Model](#) specification.

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 the SAP Web IDE for Full-Stack Development 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.

#### i 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 the [Cloud MTA Build Tool](#). Afterward, you deploy the MTA using the Cloud Foundry Command Line Interface.

**i 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

**i Note**

An MTA development descriptor `mta.yaml` is not required.

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 294]</a>
Multitarget Application archive	<a href="#">Defining Multitarget Application Archives [page 293]</a>

To learn more about	See
Multitarget Application extension descriptor	<a href="#">Defining MTA Extension Descriptors [page 329]</a>
Multitarget Application module types and parameters	<a href="#">MTA Module Types, Resource Types, and Parameters for Applications in the Cloud Foundry Environment [page 359]</a>
How to deploy the Multitarget Application	<a href="#">Multitarget Application Plug-In for the Cloud Foundry Command Line Interface [page 936]</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.
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 the WEB IDE or from Multitarget Application Archive Builder tool or manually. This file is similar to Development Descriptor but is used from the MTA Deployer.
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 301]</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.

Term	Description
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>• <b>provides:</b> indicates the properties or parameters that are provided by a module or resource to other modules.</li> <li>• <b>requires:</b> 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 Cloud Foundry deploy service:

- Maximum size of the MTA archive: 4 GB
- Maximum size of MTA module content: 1 GB
- Maximum size of MTA resource content: 1 GB
- Maximum size of MTA descriptors (`mtad.yaml` and `MANIFEST.MF`): 1 MB

## Related Information

[JAR File Specification](#) ↗

[Cloud MTA Build Tool](#) ↗

### 3.1.15.1 Benefits of the Multitarget Application Model (MTA) for the Cloud Foundry Environment

The benefits of the MTA approach can be related to the three “A”s that the MTA offers - Abstraction, Automation, and Assembly.

- Abstraction - The MTA model abstracts the underlying heterogeneity of application parts, which can grow over time, and offers a standardized way to model dependencies and environment specific configurations. In the absence of such a common model, you would have to individually handle all this heterogeneity, using custom scripts in a CI/CD tool to manage all underlying components, the service dependencies, and the configurations – achieving the needed automation through such custom scripts is not a trivial task. And maintaining such scripts – as the technologies and services used by the application changes – demands higher effort. Modeling your application as an MTA highly simplifies the lifecycle of such heterogeneous applications.

- Automation - By specifying your application structure and its dependencies (the “What”), you set the MTA Deploy Service to handle the automated deployment or undeployment, service instance creation and binding, “Blue-Green” updates, configuration injection, and so on (the “How”). This automation significantly reduces the overhead needed in CI/CD pipelines to orchestrate all the steps needed for such deployment related processes.
- Assembly - By offering a standard packaging format for your applications, the MTA serves as a release artifact. Such an assembly (of the application and all its metadata) can be useful in different contexts: in audited environments where a signed artifact is mandatory, to move deployable artifacts through a firewall, to apply different policies based on the deployment target, or to distribute the application to other consumers (for example a partner distributing a business application to several customers, who deploy it in their accounts or tenants).

## Deciding on the Right Size for an MTA

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 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 an “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 such one Cloud Foundry app in one MTA modeling still 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.

### 3.1.15.2 Getting Started

This section lists the example steps to deploy your first multitarget application. An mtar archive and an extension descriptor (optional) are used to execute the deployment.

For more information about the extension descriptor, see [Defining MTA Extension Descriptors \[page 329\]](#)

1. Copy the example below to an `mtad.yaml` file.

### ❖ Example

```
_schema-version: "3.1"
ID: app
version: 1.0.0
modules:
  - name: my-first-app
    type: staticfile
    path: content.zip
    requires:
      - name: my-first-app-service
resources:
  - name: my-first-app-service
    type: org.cloudfoundry.managed-service
    parameters:
      service: application-logs
      service-plan: lite
```

2. Create a `content.zip` archive which contains an `index.html` file.

3. Execute the following:

1. `cf add-plugin-repo CF-Community https://plugins.cloudfoundry.org`
2. `cf install-plugin multiapps`
3. `cf deploy ./`

4. Check your application

### ❖ Example

#### Output

##### ↳ Output Code

```
Deploying multi-target app archive app.mtar in org deploy-service / space
<SPACE> as <USER>...
Uploading 1 files...
  <PATH_TO_MTAR>
OK
Deploying in org "<ORG>" and space "<SPACE>"
Detected MTA schema version: "3"
No deployed MTA detected - this is initial deployment
Detected new MTA version: "2.4.0"
Processing service "my-first-app-service"...
Creating service "my-first-app-service" from MTA resource "my-first-app-
service"...
Creating application "my-first-app" from MTA module "my-first-app"...
Uploading application "my-first-app"...
Scaling application "my-first-app" to "1" instances...
Staging application "my-first-app"...
Application "my-first-app" staged
Starting application "my-first-app"...
Application "my-first-app" started and available at "deploy-service-
<SPACE>-my-first-app.cfapps.sap.hana.ondemand.com"
Skipping deletion of services, because the command line option "--delete-
services" is not specified.
Process finished.
Use "cf dmol -i 830247915" to download the logs of the process.
```

In the output example above, the application `my-first-app` is deployed and started. A service called `my-first-app-service` is also created and is bound to the application. Credentials are provisioned for the service instance and delivered to the application runtime in the `VCAP_SERVICES` environment variable.

To check the application, execute `cf apps`. The result should be as follows :

#### ↳ Output Code

```
cf apps
Getting apps in org <ORG> / space <SPACE> as <USER>...
OK
name      requested state    instances   memory   disk   urls
my-first-app  started        1/1       1G       1G     deploy-
service-<SPACE>-my-first-app.cfapps.sap.hana.ondemand.com
```

To check the service, execute `cf services`. The result should be as follows :

#### ↳ Output Code

```
cf services
Getting services in org <ORG> / space <SPACE> as <USER>...
name      service    plan      bound apps   last
operation
my-first-app-service  application-logs  lite      my-first-app  create
succeeded
```

### 3.1.15.3 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.

#### i 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 [Inside an MTA Descriptor](#).

## Related Information

[Developing Multitarget Applications](#)

### 3.1.15.4 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.

#### i 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.

An MTA archive consists of the following:

- The `MANIFEST.MF` file
- The `mtad.yaml` MTA deployment descriptor file
- Application binaries, content, or configuration files

#### i 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 Cloud Platform requests them.
- Using a `resources` directory as 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

#### i 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 Cloud Platform 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`

#### i 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

<https://sap.github.io/cloud-mta-build-tool/>

[The Multitarget Application Model](#)

[JAR File Specification](#)

[Defining MTA Deployment Descriptors for the Neo Environment](#)

[Defining MTA Extension Descriptors \[page 329\]](#)

[MTA Module Types, Resource Types, and Parameters for Applications in the Neo Environment](#)

## 3.1.15.5 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 a deployable artefacts and the MTA Deployment service in SAP Cloud Platform 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 \[page 300\]](#).

#### i Note

As there are technical similarities between SAP HANA XS Advanced and Cloud Foundry, you can adapt application parameter for operation in either platforms. Note that each environment supports its own set of module types, resource types, and configuration parameters. For more information, see the official [Multitarget Application Model](#) specification.

## 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 <https://github.com/cloudfoundry-incubator/multiapps-controller/wiki/Deployment-Descriptor#editor-schema-support>

### 3.1.15.5.1 MTA Deployment Descriptor Examples

Examples of MTA deployment descriptors.

#### i 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 Cloud Platform 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: ztana1
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

```

### i 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” use 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
```

### 3.1.15.6 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 \[page 301\]](#) - they contain the properties of module types, which represent Cloud Foundry applications or content that form the MTA and are deployed.
- [Resources \[page 318\]](#) - 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. For more information, see .
- 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 [MTA Module Types, Resource Types, and Parameters for Applications in the Cloud Foundry Environment \[page 359\]](#).
- Properties - these result in the application environment variables that have to be available to the respective module at run time. For more information, see [Parameters and Properties \[page 324\]](#).
- Technical configuration parameters, such as URLs, and application configuration parameters such as environment variables For more information, see [Parameters and Properties \[page 324\]](#).
- Metadata - provide additional information about the declared parameters and properties. For more information, see [Metadata for Properties and Parameters \[page 328\]](#).
- [Module Hooks \[page 315\]](#) - 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.
- [Alternative Grouping of MTA Properties \[page 329\]](#) - if needed, use alternative arrangement of the properties' syntax.

### 3.1.15.6.1 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`, `type`, `description`, `properties`, and `parameters`, plus `requires` and `provides` lists.

### 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
<code>javascript.no.dejs</code>	None	None	CF application with Automatic buildpack detection
<code>nodejs</code>	<code>buildpack(nodejs_buildpack)</code>	None	CF application with Node.js runtime
<code>custom</code>	None	None	CF application with Automatic buildpack detection
<code>application</code>	None	None	CF application with Automatic buildpack detection

Module Type	Default Parameter Values and Description	Module Properties	Result
java	buildpack(sap_java_buildpack)	None	CF application with Automatic runtime detection by sap_java_buildpack
java.tomcat	buildpack(sap_java_buildpack)	TARGET_RUNTIME (tomcat)	CF application with Tomcat runtime of sap_java_buildpack
java.tomee	buildpack(sap_java_buildpack)	TARGET_RUNTIME (tomee)	CF application with TomEE runtime of sap_java_buildpack
com.sap.xs.hdi	<ul style="list-style-type: none"> <li>• no-route(true) 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) In circular module-dependencies, deploy modules with dependency type "hard" first</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>i Note</b></p> <p>Before using this module type, update the content deployer applications to their latest version.</p>	None	CF application with SAP Fiori launchpad 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) For more information, see <a href="#">Tasks [page 334]</a>.</li> <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> </ul>		

Module Type	Default Parameter Values and Description	Module Properties	Result
com.sap.portal.site-content	This module type is <b>deprecated</b> . You have to use com.sap.portal.content instead.	None	CF application with SAP Fiori launchpad content
com.sap.application.content	<p>Required dependency parameters:</p> <ul style="list-style-type: none"> <li>• <code>content-target (false)</code> Specify that the resource would be used as a target for the module content deployment.</li> </ul>	None	Direct content deployment to backing services
com.sap.html5.application-content	<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 the Business Logging for configuring text resources

Module Type	Default Parameter Values and Description	Module Properties	Result
com.sap.business-logging.content	<p><b>i 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) For more information, see <a href="#">Tasks [page 334]</a>.</li> <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> </ul>		Deploys the Business Logging for configuring text resources
business-logging	<p>This module type is <b>deprecated</b>. You have to use com.sap.business-logging.content instead.</p>	None	Deploys the Business Logging for configuring text resources
staticfile	buildpack(staticfile_buildpack)	None	CF application with static file runtime
ruby	buildpack(ruby_buildpack)	None	CF application with Ruby 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
dotnet_core	buildpack(dotnet_core_buildpack)	None	CF application with Dotnet 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:

MTA Development and Deployment Module Parameters

Parameter	Read-Only (System)	Description	Default Value	Example
app-name		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 com.sap.xs2.samples.xsj shellworld.node-hello-world
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: [java_buildpack, nodejs_buildpack, staticfile_buildpack]
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

Parameter	Read-Only (System)	Description	Default Value	Example
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 com.sap.xs2.samples.xsj shellworld.node-hello-world
default-host	Yes	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.	Generated as described in the description	trial-a007007-node-hello-world
<p><b>i Note</b></p> <p>Any characters that are not within the allowed a–z and 0–9 ranges are replaced by dashes (-). This is done to prevent deployment issues.</p>				
default-instances	Yes	The number of application instances that will be started during the deployment	1	default-instances: 1
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} will be the same as \${default-domain}, unless specified explicitly	Generated as described in the description.	trial-a007007-node-hello-world.cfapps.acme.ondemand.com
default-url	Yes	The default URL, composed as \${protocol}://\${default-uri}. Note that the \${default-uri} placeholder will be resolved as \${host}.\${domain} (host-based routing)	Generated as described in the description.	\${protocol}://\${default-uri}
dependency-type		Deployment order of modules with circular dependencies	soft	dependency-type: hard 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

Parameter	Read-Only (System)	Description	Default Value	Example
docker	Write	<p>Creates a module from a docker image. When using a docker image parameter, we do not need to specify the module in the <code>MANIFEST.mf</code> 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.</p> <p>When uploading a docker image, the content of a module is not needed.</p>	n/a	<p>↳ Sample Code</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 will be available later	<code> \${default-domain}</code>	<code>domain: \${default-domain}.acme.com</code>
domains		The domains on which the application will be available later. The resulting application routes will be the Cartesian product of the domains and hosts. That is, a separate route for each host is constructed on each domain.	<code>domains: - \${default}</code>	<code>domains: - \${default-domain}.acme.com - test-\${default-domain}.acme.com</code>
enable-ssh		Enables use of SSH within an application. Supported for the Diego container runtime environment only.	n/a	<code>"enable-ssh": true</code> <code>"enable-ssh": false</code>
health-check-http-endpoint		If the <code>health-check-type</code> parameter is set to <code>http</code> , the controller will do a GET request to this endpoint. The application will be considered as healthy if the response is 200 OK.	If <code>health-check-type</code> is set to <code>http</code> , the default value is <code>/</code> , otherwise there is no default value	<code>health-check-type: ht</code> <code>health-check-http-</code> <code>endpoint: /health</code>
health-check-timeout		The application health check timeout in seconds	n/a	<code>health-check-timeout: 120</code>
upload-timeout		The application upload timeout in seconds	3600	<code>upload-timeout: 1800</code>
health-check-type		The application health check type	http	<code>health-check-type: port</code> <code>health-check-type: http</code> <code>health-check-type: process</code>

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>	\${default-host}	<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>	hosts: - \${host}	<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>	\${default-domain}	<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>	\${default-domain}	<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>	\${default-host}-idle	<pre>modules: - name: app   type: nodejs   parameters:     idle-host: "&lt;some-hostname&gt;"</pre>
<b>i 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>i 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.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>	instances: 2
keep-existing		<p>Defines the application attributes which will be kept after the deployment or blue-green deployment has finished.</p> <p>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</p>	false	<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><b>↳ Sample Code</b></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>
<p><b>i 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>cf 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>				
memory		The memory limit for all instances of an application. This parameter requires a unit of measurement M, MB, G, or GB in upper or lower case.	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>i 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: true   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 343]</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 prebuilt root file system ( <code>rootfs</code> ) you want to use.	n/a	<p><b>↳ Sample Code</b></p> <pre>modules:   - name: foo     type: application     parameters:       stack: cflinuxfs3</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>
tcps		Specifies whether the application should have TCPS type routes.	false	<code>tcps:true</code>
timestamp	Yes	Current timestamp in milliseconds	Generated as described in the description.	
zdm-mode		Parameter value of the <code>com.sap.xs.hdi</code> module type that if set to <code>true</code> , runs the deployment in a ZDM mode.	n/a	<code>zdm-mode: true</code>

## 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:

### Code Syntax

Multiple MTA Module Entries in the Deployment Descriptor (`mtad.yaml`)

#### Sample Code

```
_schema-version: "3.2.0"
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

<https://docs.cloudfoundry.org/services/managing-service-brokers.html> ↗

## 3.1.15.6.2 Module Hooks

Define and execute hooks at specific phases of module deployment.

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>	<code>task</code>	Executed before the application corresponding to the module is stopped.

Phase	Supported for types	Description
deploy.application.after-stop	task	Executed after the application corresponding to the module is stopped.
deploy.application.before-start	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: db_migration
command	Yes	Defines the actual command that is executed as a Cloud Foundry task.		command: "bin/ rails db:migrate"
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 memory: 1G
<p style="text-align: right;">→ Remember</p> <p>Do not rely on the default value, as it will probably be much more higher than you need and may not fit into the limitations of your quota.</p>				
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 disk-quota: 1G
<p style="text-align: right;">→ Remember</p> <p>Do not rely on the default value, as it will probably be much more higher than you need and may not fit into the limitations of your quota.</p>				

## 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 \[page 295\]](#)

[Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 294\]](#)

[Defining MTA Extension Descriptors \[page 329\]](#)

[Legacy Blue-Green Deployment \[page 350\]](#)

### 3.1.15.6.3 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`
- `properties`
- `parameters`
- (Optional) `active` - its value can be `true` or `false` and the default value is `true`. If set to `false`, the resource will not be processed and it will be ignored in the `requires` list of the application that requires it.

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.

### i 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.

## MTA Resource Types

### Default Resource Types

Modify the default MTA resource types by providing specific properties or parameters in the MTA deployment descriptor.

MTA Default Resource Types and Mapped Services

Resource Type	Service	Service Plan	Created Service
<code>com.sap.xs.hana-sbss</code>	<code>hana</code>	<code>sbss</code>	Service-broker security
<code>com.sap.xs.hana-schema</code>	<code>hana</code>	<code>schema</code>	Plain schema
<code>com.sap.xs.hana-securestore</code>	<code>hana</code>	<code>securestore</code>	SAP HANA secure store
<code>com.sap.xs.hdi-container</code>	<code>hana</code>	<code>hdi-shared</code>	HDI container

#### i 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
```

<code>com.sap.xs.jobscheduler</code>	<code>jobscheduler</code>	<code>lite</code>	Job Scheduler
<code>com.sap.xs.uaa</code>	<code>xsuaa</code>	<code>applicaton</code>	Application UAA
<code>com.sap.xs.uaa-application</code>	<code>xsuaa</code>	<code>application</code>	Application UAA
<code>com.sap.xs.uaa-devuser</code>	<code>xsuaa</code>	<code>application</code>	Application UAA
<code>com.sap.xs.uaa-broker</code>	<code>xsuaa</code>	<code>broker</code>	Broker UAA
<code>com.sap.xs.uaa-space</code>	<code>xsuaa</code>	<code>application</code>	Application UAA

Resource Type	Service	Service Plan	Created Service
com.sap.xs.application-logs	application-logs	lite	Streams logs of bound applications to a central application logging stack
<b>! Restriction</b>	This resource type is now deprecated. Use application-logs instead.		
com.sap.portal.site-content	portal-services	site-content	Portal services
<b>! Restriction</b>	Use only with the SAP Node.js module @sap/site-content-deployer		
com.sap.portal.site-host	portal-services	site-host	Portal services
<b>! Restriction</b>	Only for use with the SAP Node.js module @sap/site-entry		
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

Resource Type	Service	Service Plan	Created Service
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
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:

MTA Development and Deployment Parameters

Parameter	Read-Only (System)	Description	Default Value	Example
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 service name, which are combined in a way that conforms to the container-name restrictions for length and legal characters.	Generated as described in the description.	INITIAL_INITIAL_SERVICE_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
default-xsapppname	Yes	Default value for the <code>xsapppname</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>xsapppname</code> restrictions for length and legal characters.	Generated as described in the description.	xs-deploy-service-database (if the service name is "xs@-deploy-service-database")
service		The type of the created service	Empty, or as specified in <code>resource-type</code>	service: hana
service-alternatives		List of alternatives of a default service offering, defined in the deploy service configuration. If a default service offering does not exist for the current org/space or creating a service to it fails (with a specific error), service alternatives are used. The order of service alternatives is considered.	Empty, or as specified in <code>in the deploy service configuration (resource-type)</code>	For Coud Foundry Trial, "hanatrial" is available instead of "hana". service-alternatives: ["hanatrial"]
service-key-name	Write	Used when consuming an existing service key. Specifies the name of the service key. See Consumption of existing service keys for more information.	The name of the resource.	service-key-name: my-service-key

Parameter	Read-Only (System)	Description	Default Value	Example
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>	nodejs-hdi-container com.sap.xs2.samples.xsj shellworld.nodejs-hdi-container
service-plan		The plan of the created service	Empty, or as specified in resource-type	service-plan: hdi-shared
service-tags	Write	Some services employ a list of custom tags, which provide an easier way for applications to parse <VCA_PSERVICES> for credentials. You can provide custom tags when creating a service instance. For more information, see <a href="#">Service Tags [page 340]</a> .	n/a	<pre>resources:   - name: nodejs-uaa     type: com.sap.xs.uaa     parameters:       service-tags: ["custom-tag-A", "custom-tag-B"]</pre>
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>

### i Note

For a better understanding of the interactions among the service, service broker, and service instances, see section "[Architecture and Terminology](#)".

## Related Information

[Services Overview](#)

[Managing Services](#)

## 3.1.15.6.4 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 deployment description (`mtad.yaml`). More often, however, property values are determined during deployment, where the values are either explicitly set by the administrator, for example, in an deployment-extension descriptor file (`myDeployExtension.mtaext`). When set, the deployer injects the property values into the module's environment. The deployment operation reports an error if it cannot determine a value for a property that is referenced in another or is marked as mandatory.

#### → Tip

It is possible to declare metadata for properties defined in the MTA deployment description; the mapping is based on the parameter or property keys. For example, you can specify if a property is **required** (`optional: false`) or can be modified `overwritable: true`.

## Parameters

Parameters are reserved variables that affect the behavior of the MTA-aware tools, such as the deployer.

#### i Note

Both parameters and properties may have literal values, that is, strings, integers, and so on. This also applies to deeply nested structured values, such as arrays or maps.

Parameters can be “system”, “write-only”, or “read-write” (default value can be overwritten). Each tool publishes a list of system parameters and their (default) values for its supported target environments. All parameter values can be referenced as part of other property or parameter value strings. To reference a parameter value, use the placeholder notation `$(<parameter>)`. The value of a system parameter cannot be changed in descriptors. Only its value can be referenced using the placeholder notation.

Examples of common read-only parameters are `user`, `default-host`, `default-uri`. The value of a writable parameter can be specified within a descriptor. For example, a module might need to specify a non-default value for a target-specific parameter that configures the amount of memory for the module's runtime.

#### → 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`.

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.

#### ≡, Sample Code

##### Parameters and Placeholders

```
modules:  
  - name: node-hello-world  
    type: javascript.nodejs
```

```
path: web/
parameters:
  host: ${user}-node-hello-world
```

### → Tip

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.

## Properties

The MTA deployment descriptor can contain two types of properties, which are very similar, and are intended for use in the `modules` or `resources` configuration, respectively.

Properties can be declared in the deployment description both in the `modules` configuration (for example, to define `provides` or `requires` dependencies), or in the `resources` configuration to specify `requires` dependencies. Both kinds of properties (`modules` and `requires`) are injected into the module's environment. In the `requires` configuration, properties can reference other properties that are declared in the corresponding `provides` configuration, for example, using the `~{}` syntax.

## Dependency-Specific Parameters

These parameters can be used with the `provides` or `requires` dependencies:

### Dependency - Specific Parameters

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

Parameter	Scope	Read-Only (System)	Description	Default Value	Example
visibility	provided dependency	Write	Specifies the organizations and spaces in which public provided dependencies are visible. See <a href="#">Visibility of cross-MTA configuration</a> for more information.	In all spaces of the current organization.	<pre>visibility:   - org: foo     space: "*"   - org: bar     space: "*"   - org: baz     space: qux</pre> <pre>visibility:   -     org: \${org}     space: "*"</pre>

As an alternative, you can also externalize such configurations in a file. See [Service Creation Parameters \[page 337\]](#).

## 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

```
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.

### 3.1.15.6.5 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

```

### 3.1.15.6.6 Alternative Grouping of MTA Properties

The deploy 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

```

### 3.1.15.7 Defining MTA Extension Descriptors

The Multitarget Application (MTA) extension descriptor is a YAML file that contains data complementary to the deployment descriptor. The data can be environment or deployment specific, for example, credentials depending on the user who performs the deployment. The MTA extension descriptor is a YAML file that has a

similar structure to the deployment descriptor, by following the Multitarget Application Model structure with several limitations and differences. Normally, extension descriptor extends deployment descriptor but it is possible to extends other extension descriptor, making extension descriptors chain.. It can add or overwrite existing data if necessary.

Several extension descriptors can be additionally used after the initial deployment.

#### i 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 Cloud Platform which MTA specification version should be used. Furthermore, the schema version used within the extension descriptor and the deployment descriptor should always be same.

In the examples below, we have a deployment descriptor, which has already been defined, and several extension descriptors.

#### i Note

Each extension descriptor is defined in a separate file with an extension .mtaext.

Deployment descriptor:

#### • Example

```
_schema-version: '3.1'  
parameters:  
  hcp-deployer-version: '1.0'  
ID: com.example.extension  
version: 0.1.0  
resources:  
  - name: data-storage  
    properties:  
      existing-data: value
```

The example above instructs SAP Cloud Platform to:

- Validate the extension descriptor against the MTA specification version 3.1
- Extend the com.example.extension deployment descriptor

The following is a basic example of an extension descriptor that adds data and overwrites data to another extension descriptor:

#### • Example

```
_schema-version: '3.1'  
ID: com.example.extension.first  
extends: com.example.extension  
resources:  
  - name: data-storage  
    properties:  
      existing-data: new-value  
      non-existing-data: value
```

The above instructs SAP Cloud Platform to:

- Extend the deployment descriptor by its ID `com.example.extension`
- Validate the extension descriptor against the MTA specification version 3.1
- Overwrite the data for the `existing-data` property
- Add a new data called `non-existing-data` to the `data-storage` properties

The following is an example of another extension descriptor that extends the extension descriptor from the previous example:

### ❖ Example

```
_schema-version: '3.1'  
ID: com.example.extension.second  
extends: com.example.extension.first  
resources:  
  - name: data-storage  
    properties:  
      second-non-existing-data: value
```

The example above instructs the SAP Cloud Platform to:

- Extend the first extension descriptor by its ID
- Add a new data called `second-non-existing-data` to the `data-storage` properties
- The examples above are incomplete. To deploy a solution, you have to create a deployment descriptor and an MTA archive.

### What is possible to do with an extension descriptor?

You can do the following using an extension descriptor:

- Add a new data in modules, resources, parameters, properties, provides, requires sections
- Overwrite an existing data (in depth) in modules, resources, parameters, properties, provides, requires sections
- 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 \[page 328\]](#).

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

## Related Information

[Defining MTA Deployment Descriptors for the Neo Environment](#)

[Defining Multitarget Application Archives \[page 293\]](#)

[MTA Module Types, Resource Types, and Parameters for Applications in the Neo Environment](#)

[The Multitarget Application Model](#) 

## 3.1.15.8 Cloud Platform Capabilities

This section contains information about how to manage standard Cloud Foundry entities with MTA modelling.

- [Applications \[page 332\]](#)
- [Tasks \[page 334\]](#)
- [Services \[page 335\]](#)
- [Routes \[page 343\]](#)

### 3.1.15.8.1 Applications

If you want to create an application in Cloud Foundry and use the MTA deployment service, you must create a module first. In the deployment descriptor the module represents an application. These modules can have different types and parameters which modify their behavior.

#### 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`, and then the set of operations to change the current state of the MTA to the desired state are computed. The following optimizations are possible during the MTA redeployment:

- The application bits are reuploaded if:
  - 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 application attributes, such as commands, buildpacks, memory, health-check-url, and so on, are changed
  - The application environment is changed
  - There are new services the application is bound to
  - There are services the application is bound to, which are discontinued (not available) in the new version of the MTA

#### i Note

Currently, Cloud Foundry environment applications are restaged if they are bound to any service with parameters, because it is not possible to get the current Cloud Foundry service parameters and compare them with the desired ones.

- The application is bound to service configuration parameters that 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.

## 3.1.15.8.2 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.

### i 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 do not 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.
- Only when you create an MTA development descriptor (`mta.yaml`) to build with the Cloud MTA Build Tool, you have to enter the parameter `no-source: true` in the `build-parameters` section.

## 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](#)

### 3.1.15.8.3 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.

#### i 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 \[page 295\]](#).

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

<https://docs.cloudfoundry.org/devguide/using-tasks.html>

### 3.1.15.8.4 Services

If you want to create a service in the Cloud Foundry environment using the MTA 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 certain combination of service offering and plan. See subsection “MTA resource types” in [Resources \[page 318\]](#).

For more flexible approach use the resource types listed below.

#### i 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 \[page 318\]](#) for more information.

#### Special Resource Types

- `org.cloudfoundry.managed-service`

To choose a managed service and/or service plan that is not listed in [MTA Resource Types \[page 320\]](#), you define it using the `org.cloudfoundry.managed-service` resource type with the following parameters:

- `service`

**Required.** Name of the service to create.

- `service-name`

**Optional.** Service instance name. Default value is the resource name.

- `service-plan`

**Required.** Name of the service to create.

For example:

#### « Sample Code

```

resources:
  - name: my-postgre-service
    type: org.cloudfoundry.managed-service
    parameters:

```

SAP Cloud Platform  
Development

PUBLIC 335

```
service: postgresql
service-plan: v9.6-dev
```

## i 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 with:

### ↳ Sample Code

```
resources:
  - name: my-postgre-service
    type: org.postgresql
    parameters:
      service-plan: v9.6-dev
```

- `org.cloudfoundry.existing-service`

To assume that the named service exists, but to not manage its lifecycle, you define the service name by using the `org.cloudfoundry.managed-service` resource type with the following parameters:

- `service-name`

**Optional.** 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 \[page 341\]](#).

- `org.cloudfoundry.user-provided-service`

Create or update a user-provided service configured with the following resource parameters:

- `service-name`

**Optional.** Name of the service to create. Default value is the resource name.

- `config` **Required.** 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 \[page 344\]](#).

### 3.1.15.8.4.1 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

#### i 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> <p>MTA deployment descriptor or extension</p> <pre>resources:   - name: java-uaa     type: com.sap.xs.uaa     parameters:       config:         xsappname: java-hello-world         path: &lt;path to directory&gt;</pre>	<p>↳ Sample Code</p> <p>MTA deployment descriptor or extension</p> <pre>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:       config:         xsappname: java-hello-world         path: &lt;path to directory&gt;</pre>
	<p>↳ Sample Code</p> <p>xs-security.json</p> <pre>{   "xsappname": "java-hello-world" }</pre>	<p>↳ Sample Code</p> <p>xs-security.json</p> <pre>{   "xsappname": "java-hello-world" }</pre>
	<p>↳ Sample Code</p> <p>Additional entry in MANIFEST.MF</p> <pre>Name: xs-security.json MTA-Resource: java-uaa Content-Type: application/json</pre>	<p>↳ Sample Code</p> <p>Additional entry in MANIFEST.MF</p> <pre>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

[HTML5 Application Repository Router](#)

[Managing Service Instances with the CLI](#)

### 3.1.15.8.4.2 Service Binding Parameters

Some services support additional configuration parameters in the `create-bind` request; these parameters are passed in a valid JSON object containing the service-specific configuration parameters.

The deployment service supports the following methods for the specification of service-binding parameters:

- Method 1 - An entry in the MTA deployment descriptor (or the extension)
- Method 2 - A JSON file with the required service-configuration parameters

i 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>modules:   - name: node-hello-world-backend     type: javascript.nodejs     requires:       - name: node-hdi-container     parameters:       config:</pre> <p>permissions: debugging</p>	<p>↳ Sample Code</p> <p>MTA deployment descriptor or extension</p> <pre>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: debugging</p>
	<p>↳ Sample Code</p> <p>Content of xs-hdi.json</p> <pre>{   "permissions": "debugging" }</pre>	<p>↳ Sample Code</p> <p>Content of xs-hdi.json</p> <pre>{   "permissions": "debugging" }</pre>
	<p>↳ Sample Code</p> <p>Additional entry in MANIFEST.MF</p> <pre>Name: xs-hdi.json MTA-Requires: node-hello-world-backend/ node-hdi-container Content-Type: application/json</pre>	<p>↳ Sample Code</p> <p>Additional entry in MANIFEST.MF</p> <pre>Name: xs-hdi.json MTA-Requires: node-hello-world-backend/ node-hdi-container Content-Type: application/json</pre>

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, 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.

### i 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>`.

### 3.1.15.8.4.3 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 deployment service about custom tags, you can use the special `service-tags` parameter, which 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"]
```

#### i Note

Some service tags are inserted by default, for example, `xsuaa`, for the XS User Account and Authentication (UAA) service.

### 3.1.15.8.4.4 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-name: jobscheduler
    service-broker-user: ${generated-user}
    service-broker-password: ${generated-password}
    service-broker-url: ${default-url}
```

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.

#### i Note

During the creation of the service broker, the XS advanced 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.

#### i 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.

### 3.1.15.8.4.5 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`.

### i 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.

## 3.1.15.8.5 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.

### i Note

A `routes` parameter consists of one or many HTTP routes following the pattern `myhost.my.domain/path`.

### ❖ 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"
```

## 3.1.15.9 Features

The section describes MTA-specific features available in SAP Cloud Platform Cloud Foundry environment.

Each feature has its own productive benefit. For example, the blue-green deployment provides zero downtime update of your MTA.

- [Cross-MTA Dependencies \[page 344\]](#)
- [Blue-Green Deployment of Multitarget Applications \[page 349\]](#)
  - [Legacy Blue-Green Deployment \[page 350\]](#)
  - [Blue-Green Deployment Strategy \[page 352\]](#)
- [Order of Deployment \[page 354\]](#)
- [Features Related to SAP Cloud Platform Alert Notificaton \[page 355\]](#)
- [Content Deployment \[page 356\]](#)

- Docker Images as Part of an MTA Deployment [page 333]

### 3.1.15.9.1 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} # Wildcard * searches in all spaces

```

### → 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. If you specify a wildcard value (\*) for organization or space of the provider MTA, the provider would be searched in all organization or spaces for which the wildcard value is provided.

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.

### i 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.

### 3.1.15.9.1.1 Plugins

The deployment service supports a method that allows an MTA to consume multiple configuration entries per `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 deploy service will attempt 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.

#### i 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) contributes additional configuration details after subscriber applications have been deployed, the subscriber applications will not receive the new information immediately; they will be made aware of the new configuration details only when they are updated. Note, however, that the update operation will fail because multiple configuration entries will be 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 will be 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.

### i 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. One consequence of this

registration is that if any new configuration entries are published in the configuration registry during the deployment of another MTA, and those new entries match the filter specified in the subscription of an application, then that application's environment would be updated, and the application itself would be restarted in order for it to see its new environment's state.

→ Tip

When starting the deployment of an MTA (with the `xs 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.

### 3.1.15.9.2 Blue-Green Deployment of Multitarget Applications

Run two identical production environments to employ the blue-green deployment technique.

! 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 \[page 350\]](#) - where the productive environments are called "blue" and "green"
- [Blue-Green Deployment Strategy \[page 352\]](#) - where the production environments are called "live" and "idle".

### Related Information

<https://docs.cloudfoundry.org/devguide/deploy-apps/blue-green.html> ↗

### 3.1.15.9.2.1 Legacy Blue-Green Deployment

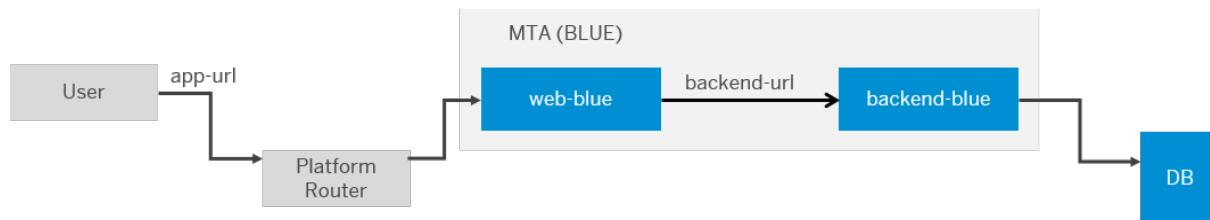
Use the legacy blue-green deployment strategy of Multitarget applications.

#### 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:



#### Procedure

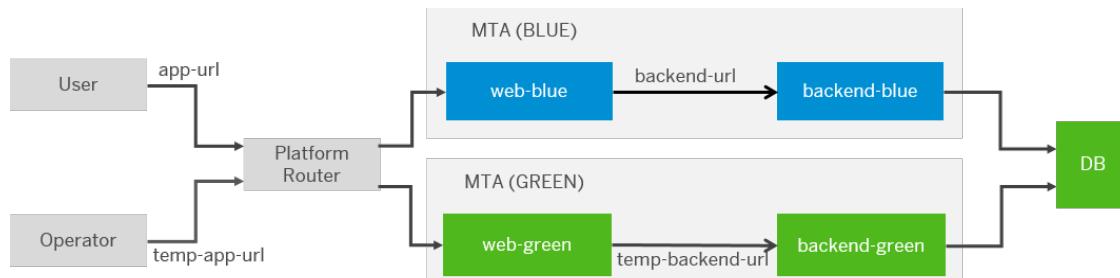
1. Deploy your updated MTA (the green version) by executing the `cf bg-deploy <your-mta-archive-v2>` command.

##### i Note

The first action is that all MTA services are updated. The changes between the old and the new versions must be compatible, for example, between the old and the new versions of database tables, UAA configurations, and so on.

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
```

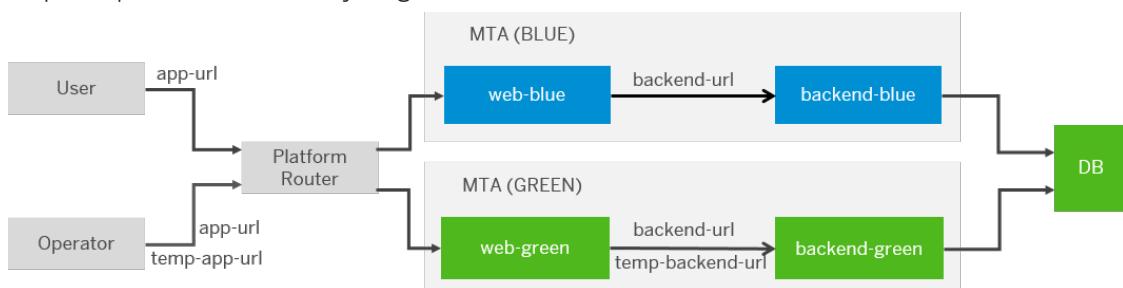
2. Optionally, test the “green” version using the temporary routes.

If you do not want to make the “green” version available, stop the process. Note that this does not automatically remove the green versions and the temporary routes.

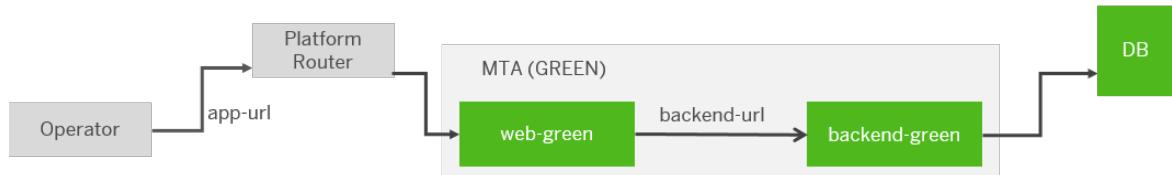
3. To make the “green” version productive, choose [Resume](#).

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



## Related Information

[Blue-Green Deployment of Multitarget Applications \[page 349\]](#)

[Blue-Green Deployment Strategy \[page 352\]](#)

### 3.1.15.9.2.2 Blue-Green Deployment Strategy

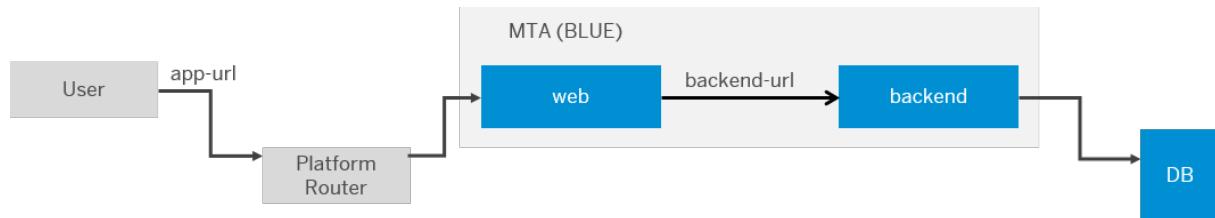
Use the current blue-green deployment of Multitarget applications.

#### 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.
- Currently this feature is not compatible with partial deployment of applications, for example, when you use `-m` or `-r` with the deployment command.
- Do not use the parameter `$(app-name)` when using the blue-green deployment, as it might cause errors connected with the temporary application name.

You have a previously deployed MTA, with functional productive applications and routes:



#### Context

##### i Note

If you already have executed deployments using the `cf bg-deploy` command and they have been successful, you can switch to the `deploy --strategy blue-green` deployment method, or vice versa. Do not begin a blue-green deployment using a given deployment command and continue with the other, as this might result in downtime.

#### Procedure

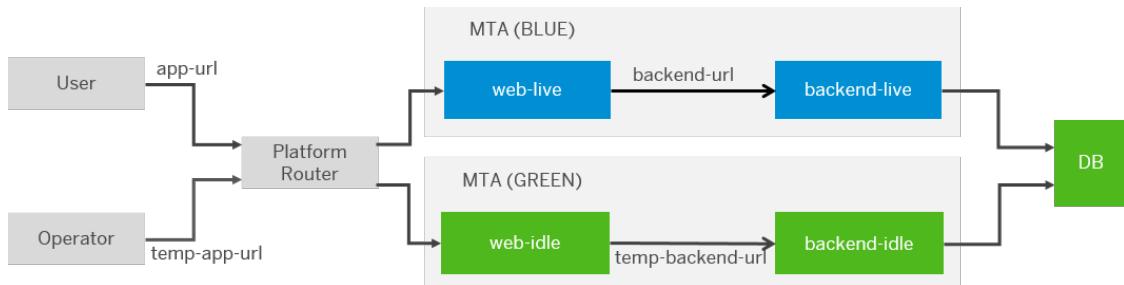
1. Deploy your updated MTA in idle state by executing the command `cf deploy <your-mta-archive-v2> --strategy blue-green`.

## i Note

The first action is that all MTA services are updated. The changes between the old and the new versions must be compatible. For example, the changes between the old and the new versions of the database tables, UAA configurations, and so on.

This creates:

- new applications adding “idle” to the original application names
- 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. You can skip this step using the command line option **--skip-testing-phase**.
3. If you do not want to make the idle version available, abort the process. Note that this automatically removes the idle versions and the temporary routes. If you want to make it 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

You can see an example of the process at [Blue-Green Deployment](#) ↗

## Related Information

[Blue-Green Deployment at GitHub](#) ↗

[Blue-Green Deployment of Multitarget Applications](#) [page 349]

[Legacy Blue-Green Deployment](#) [page 350]

### 3.1.15.9.3 Order of 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

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.

## 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.

### 3.1.15.9.4 Features Related to SAP Cloud Platform Alert Notification

Deploy and update the SAP Cloud Platform Alert Notification service 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 Alert Notification. You can use the same procedure to update the service parameters. See the procedure described in [Updating for the Cloud Foundry Environment](#), section *Update using a Multi-Target Application deployment descriptor*.

## Using Alert Notification for alerts during deployment and undeployment of multitarget applications

### 3.1.15.9.5 Content Deployment

Direct content deployment provides a mechanism to deploy content to services without the need of an application-specific deployer application.

#### i Note

It is supported only for schema version 3.1 and higher.

To do this, you have to use the module type `com.sap.application.content`, which requires a dependency to a service or an existing service key, where the required dependency parameter `content-target` is set to `true`.

Cloud Foundry services currently supporting GACD are:

- `html5-apps-repo`
- `workflow`

Both of them are modeled as GACD deployment by WEB IDE when creating the respective modules.

#### ❖ Example

An example of a managed-service that creates a workflow service with plan `standard`, and then uses it to deploy the content from a module type `content-module`:

```
_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
```

You can also customize the service keys created as part of the content deployment by using the required dependency parameter `service-key`.

#### ❖ Example

An example for customizing service keys:

```
_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
  service-key:
    name: workflow_service-key
    config:
      <map entries for creation parameters>
- name: xsuaa_service
parameters:
  service-key:
    name: xsuaa_service-key
    config:
      <map entries for creation parameters>

resources:
- name: workflow_service
  type: org.cloudfoundry.managed-service
  parameters:
    service-plan: standard
    service: workflow
- name: xsuaa_service
  type: org.cloudfoundry.managed-service
  parameters:
    service-plan: application
    service: xsuaa

```

If you want to use an already existing service key, create it beforehand.

### • Example

An example of how to deploy content directly to an existing service key with a descriptor modeled as such:

```

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_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

```

You can provide additional configurations as part of the parameters of the module::

### • Example

An example for providing additional configurations as part of the parameters of the module::

```

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
parameters:
  config:
    <...>
resources:
- name: workflow_service
  type: org.cloudfoundry.managed-service
  parameters:
    service-plan: standard
    service: workflow

```

In some cases the content that is deployed might require multiple services where the `content-target` service requires credentials of other services. In order to model such a dependency you need a descriptor similar to the following:

### ❖ Example

```

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.uaa

```

In the example above, `html5_service` would gain access credentials to `xsuaa_service`.

You can also enter inline the content definition of a module as part of the parameters of the module:

### ❖ Example

#### ↳ 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-name: destination-service
  service-plan: lite
```

## Related Information

[Deploy Content Using Generic Application Content Deployer \[page 69\]](#)

### 3.1.15.10 MTA Module Types, Resource Types, and Parameters for Applications in the Cloud Foundry Environment

This section contains information about the supported MTA modules, their default parameters, properties, and supported resource types available in the Cloud Foundry environment.

#### MTA Module Types

For more information, see [MTA Module Types \[page 301\]](#)

#### MTA Resource Types

##### Default Resource Types

For more information, see [MTA Resource Types \[page 320\]](#)

##### Special Resource Types

For more information, see [Special Resource Types \[page 335\]](#)

## Parameters

Module, resource, and dependency 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:

- [Module - Specific Parameters \[page 305\]](#)
- [Resource - Specific Parameters \[page 322\]](#)
- [Module Hooks - Specific Parameters \[page 316\]](#)
- [Dependency - Specific Parameters \[page 326\]](#)
- Generic parameters that can have the following scopes:
  - Global - can be defined on root document level
  - All - can be consumed everywhere throughout the document

### MTA Development and Deployment Parameters

Parameter	Scope	Read-Only (System)	Description	Default Value	Example
authorization-url	All	Yes	The authorization URL as specified in the cloud controller's /v2/info endpoint.	Generated as described in the description.	<code>https://login.cf.sap.hana.ondemand.com</code> <code>https://localhost:30032/uaa-security</code>
controller-url	All	Yes	The URL of the cloud controller	Generated as described in the description.	<code>https://api.cf.sap.hana.ondemand.com</code> <code>https://localhost:30030</code>
default-domain	All	Yes	The default domain (configured in the Cloud Foundry environment)	Generated as described in the description.	<code>accra6024.cfapps.acme.com</code>
deploy-url	All	Yes	The deploy service URL for the Cloud Foundry environment	Generated as described in the description.	

Parameter	Scope	Read-Only (System)	Description	Default Value	Example
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>

### i 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 (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

## Related Information

[Deploy a Multitarget Application \(with Command-Line Tools\)](#)

[Multitarget Application Commands for the Cloud Foundry Environment \[page 938\]](#)

[Diego Components and Architecture](#) ↗

## 3.1.15.11 Transporting Multitarget Applications in Cloud Foundry using CTS+

You can enable transport of SAP Cloud Platform applications and application content that is available as Multitarget Applications (MTA) using the Enhanced Change and Transport System (CTS+).

## Prerequisites

- You have configured your SAP Cloud Platform subaccounts for transport with CTS+ as described in [How To... Configure SAP Cloud Platform Cloud Foundry 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 \[page 286\]](#).

## Context

You use the Change and Transport System (CTS) of ABAP to transport and deploy your applications running on SAP Cloud Platform 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 Cloud Platform application:

## Procedure

1. Package the application in a Multitarget Application (MTA) archive using the Archive Builder Tool as described in [Defining Multitarget Application Archives \[page 293\]](#).
2. Attach the MTA archive to a CTS+ transport request as described in [How To... Configure SAP Cloud Platform Cloud Foundry for CTS](#).
3. Trigger the import of an SAP Cloud Platform application as described in [How To... Configure SAP Cloud Platform Cloud Foundry for CTS](#).

## Related Information

[Resources on CTS+ !\[\]\(a8a6f0f5a901c900d78491b563b891c6\_img.jpg\)](#)

[Setting up a CTS+ enabled transport landscape in SAP Cloud Platform !\[\]\(6f5ef306cff1f7e2ad39ed5d1894731d\_img.jpg\)](#)

## 3.1.15.12 Frequently Asked Questions

### My MTA deployment failed. What could be done?

Check the [Deployment Failed \[page 367\]](#) section.

### What to do when my application start-up/staging failed?

Check the [Application Start-up Failed \[page 370\]](#) section.

### How can I find the logs of an application from a failed/succeeded deployment?

There are several ways to find the application logs:

- Use the `cf dmol -i <operation-id>` command in order to download the logs of the deployment of an operation with id `<operation-id>`. The id of the operation could be obtained using the `cf mta-ops` command. In the downloaded deployment logs, there will be a `<application-name>.log` file which contains the logs of the application.

- Use `cf logs <application-name> --recent` in order to retrieve the recent logs of the application.

**i Note**

The recent logs are available for only 10 minutes.

For more information, check the options for downloading the deployment logs in the [Deployment Failed \[page 367\]](#) section.

### How to find more information about the problematic deployment?

Use the `cf dmol -i <operation-id>` command in order to download the deployment operation logs as described in the [Deployment Failed \[page 367\]](#) section. After downloading the deployment logs, locate the file with name `MAIN_LOG` which contains the whole logs of the deployment. You will find a detailed error message in it.

### 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 \[page 367\]](#).

### 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 the MTA with the provided ID.

### What can I do with my deployment?

- If the deployment fails, see [Deployment Failed \[page 367\]](#)
- 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 [Deployment Failed \[page 367\]](#) 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 staring the deployment by providing the option `-f` as described in the [deploy \[page 939\]](#) 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 \[page 289\]](#)

## What to do in case of service creation failure?

See [Service Create/Update/Delete Failures \[page 368\]](#)

## How to make my deployment faster?

You can use a parallel deployment as described in section [Parallel Deployment \[page 355\]](#)

## How can I resolve problems with the MTA descriptor modeling?

For more information, see [Defining Multitarget Application Deployment Descriptors for Cloud Foundry \[page 294\]](#)

## 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 [Order of Deployment \[page 354\]](#).

## 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 \[page 338\]](#) as well as [Service Creation Parameters \[page 337\]](#).

## How to make sure that my archive is signed correctly by SAP and its content has not been changed ?

Use the deployment option `--verify-archive-signature` as described in [Multitarget Application Commands for the Cloud Foundry Environment \[page 938\]](#).

### 3.1.15.13 Troubleshooting

This section contains information about the following problems that may occur during the Multitarget Application deployment:

- Deployment Failed [page 367]
- Service Create/Update/Delete Failures [page 368]
- Route Mapping Failure [page 368]
- Application Binding to Service Failure [page 369]
- Application Content Upload Failure [page 369]
- One-off Tasks Execution Failure [page 370]
- Application Start-up Failed [page 370]

#### Deployment Failed

There might be different reasons for a deployment failure. This section describes the basic steps you should perform in order to recover from a failed deployment. To interact with the failed deployment, you can execute the following actions:

- **Abort** – terminates a deployment with a given operation id.

##### i Note

This action will not roll back all applied changes. 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.

Command: `cf <deployment-action> -i <process-id> -a abort`

##### ❖ Example

```
cf bg-deploy -i 12345 -a abort
```

- **Retry** – retries the last failed step(s) of a deployment with the given operation id.

Command: `cf <deployment-action> -i <process-id> -a retry`

##### ❖ Example

```
cf undeploy -i 12345 -a retry
```

- **Download deployment operation logs** – downloads the logs for the current deployment. The logs contain the following structure of files:

- MAIN\_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

Command: `cf dmol -i <process-id>`

## • Example

```
cf dmol -i 12345
```

- **Download or stream single CF application log** – in order to debug issues relevant only to a single cf application that is part of an MTA, the following cf primitives can be used:
  - `cf logs <application-name>`: streams the logs of the application as it is getting handled by the platform (stating/starting/jobs and so on)
  - `cf logs <application-name> --recent`: outputs the recent history of an application, which may contain errors preventing staging or starting of the application.

Semantics of the commands:

`<deploy-action>` - it may be deploy or bg-deploy or undeploy

`<process-id>` - the process id of the failed deployment. It can be taken from the result of the execution of the `cf mta-ops` command.

## Service Create/Update/Delete Failures

If a service operation 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>
```

Usually, such errors are produced when there is a problem with the services provisioning infrastructure. To check if there is a problem with the service, perform the following manual steps :

1. `cf create-service <service-plan> <service-offering> <service-instance-name>` - for creating the service
2. `cf update-service <service-name> [additional arguments if used]` - for updating the service
3. `cf delete-service <service-name> -f` – for deleting the service.

### i Note

This may result in data loss in case the backing service persists state.

## Route Mapping Failure

If a route could not be mapped to an application and the deployment fails, the following checks can be performed:

1. `cf routes` – displays all the created routes. Verify that the route does not exist.
2. `cf map-route <application-name> <domain> [ADDITIONAL OPTIONS]` – maps a route to the application with the given name. If the process finishes successfully, retry the deployment operation.
3. `cf unmap-route <application-name> <domain> [ADDITIONAL OPTIONS]` – removes a route for the application with the given name. If this step is successful, execute step 2.

## 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. The following steps might help to investigate the issue and eventually resolve it:

1. Retry the deployment process – for more information about actions related with the deployment, see [Deployment Failed \[page 367\]](#) section.
2. If the deployment fails after it is retried, you can try to unbind/bind the application to the service manually, using the commands:
  1. `cf bind-service <application-name> <service-name> [ADDITIONAL OPTIONS]` - binds the application with the given name to service.
  2. `cf unbind-service <application-name> <service-name>` - unbinds the application with the given name from service.

## Application Content Upload Failure

Such errors can occur in the following case::

- There is a problem with the Cloud Controller. The Cloud Controller is responsible for taking the application binaries and storing them, so that it executes the operations stage and starts it afterwards. The status code and the response returned from the Cloud Controller of such errors are in the following format:

```
400 Bad Request, Request entity too large.
```

- The application archive size is bigger than 1GB in size. This limitation is set by the Cloud Platform and could not be modified.
- There is a problem with the processing of the application content by the deployer. If this is the case, then an incident to the following component should be created:  
BC-XS-SL-DS

When such an error occurs, there are two possible workarounds:

- Retry the deployment process – for more information about actions related with the deployment, see [Deployment Failed \[page 367\]](#) section.
- Execute `cf push` instead of `cf deploy` using only the application with the problematic content.

## One-off Tasks Execution Failure

These errors happen when there are one-off tasks defined for some application and their execution fail. Usually, they fail because:

- There is a problem with the command of the one-off task  
You have to review and fix the command, which the one-off task is executing
- There is a problem with a cloud platform component responsible for the execution of the one-off task  
You have to retry the deployment process – for more information about actions related with the deployment, see [Deployment Failed \[page 367\]](#) section.

## Application Start-up Failed

Usually, this error happens when there is a problem in the application code. It should be resolved by the developer of the MTA.

In order to locate the problem in the application start-up, the logs of the application should be checked.

You can download the logs as described in the [Deployment Failed \[page 367\]](#) section.

### 3.1.16 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 \[page 371\]](#)
- [Binding Service Instances to Applications \[page 376\]](#)
- [Creating Service Instances \[page 372\]](#)
- [Creating Service Keys \[page 379\]](#)
- [Creating User-Provided Service Instances \[page 374\]](#)
- [Deleting Service Instances \[page 381\]](#)
- [Updating Service Instances \[page 383\]](#)

## Service Management

You can also access services for the Cloud Foundry environment through Service Management, the central registry for service brokers and platforms. It's tightly integrated with SAP Cloud Platform and allows you to consume services in connected runtime environments and to manage platforms, service brokers, service instances, and service bindings.

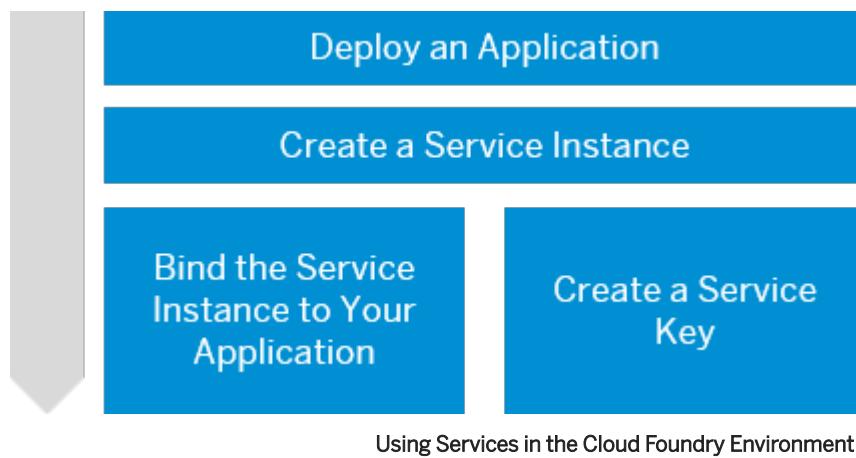
- For more information, see [Service Management Overview](#).
- To find out if Service Management is available in your desired region, check out the [Regions and Services Portfolio](#).

### 3.1.16.1 About Services

In the Cloud Foundry environment, you usually enable services by creating a service instance using either the SAP Cloud Platform 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 Cloud Platform. 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>.

#### i 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 \[page 374\]](#).

For more conceptual information about using services in the Cloud Foundry environment, see <https://docs.cloudfoundry.org/devguide/services/>.

## Related Information

[Creating Service Instances \[page 372\]](#)

[Binding Service Instances to Applications \[page 376\]](#)

[Creating Service Keys \[page 379\]](#)

[Creating User-Provided Service Instances \[page 374\]](#)

[Services and Capabilities](#)

### 3.1.16.2 Creating Service Instances

Use the SAP Cloud Platform cockpit or the Cloud Foundry Command Line Interface to create service instances:

- [Create Service Instances Using the Cockpit \[page 372\]](#)
- [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 373\]](#)

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 \[page 337\]](#)

#### 3.1.16.2.1 Create Service Instances Using the Cockpit

You can use the cockpit to create service instances.

### Prerequisites

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 \[page 781\]](#).

### Procedure

1. Navigate to the space in which you want to create a service instance.

#### i Note

We have designed new cockpit screens to view and manage service instances.

For more information, see [Creating Service Instances in Cloud Foundry](#).

2. In the navigation area, choose  [Services](#)  [Service Marketplace](#).
- All services available to you appear.
3. Choose the service for which you want to create an instance.
4. In the navigation area, choose [Instances](#).
5. Choose [New Instance](#).
6. Choose a service plan from the dropdown list, then choose [Next](#).
7. (Optional) Specify a JSON file or specify parameters in the JSON format, then choose [Next](#).
8. (Optional) If you've already deployed an application that you want to bind to the new service instance, choose it from the list. Choose [Next](#).
9. Enter a name for your instance and choose [Finish](#).

### 3.1.16.2.2 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- 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 \[page 781\]](#).

#### 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 \[page 376\]](#)

[Creating User-Provided Service Instances \[page 374\]](#)

[About Services \[page 371\]](#)

### 3.1.16.3 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 Cloud Platform cockpit or the Cloud Foundry Command Line Interface:

- [Create User-Provided Service Instances Using the Cockpit \[page 374\]](#)
- [Create User-Provided Service Instances Using the Cloud Foundry Command Line Interface \[page 375\]](#)

For more information on user-provided service instances, see <https://docs.cloudfoundry.org/devguide/services/user-provided.html>.

#### 3.1.16.3.1 Create User-Provided Service Instances Using the Cockpit

Use the cockpit to create user-provided service instances and bind them to applications in the Cloud Foundry environment.

## Context

Use user-provided services to provide an application with a URL, port, and credentials, required to authenticate the application when communicating with a service not available in the service marketplace.

## Procedure

1. Navigate to the space in which you want to create a user-provided service instance. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).

### i Note

We have designed new cockpit screens to view and manage service instances.

For more information, see [Creating User-Provided Service Instances in Cloud Foundry](#).

2. In the navigation area, choose ► *Services* ► *User-Provided Services* ▾.
3. Choose *New Instance*.
4. Enter a name for your new service instance.
5. Enter the service credentials as JSON and save your changes.

## Next Steps

To bind your application to the user-provided service instance, follow the steps described in [Bind Service Instances to Applications Using the Cockpit \[page 377\]](#).

### 3.1.16.3.2 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- 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 \[page 378\]](#).

## Related Information

[Creating Service Instances \[page 372\]](#)

[Creating Service Keys \[page 379\]](#)

[About Services \[page 371\]](#)

## 3.1.16.4 Binding Service Instances to Applications

Use the SAP Cloud Platform cockpit or the Cloud Foundry Command Line Interface to bind service instances to applications:

- [Bind Service Instances to Applications Using the Cockpit \[page 377\]](#)
- [Bind Service Instances to Applications Using the Cloud Foundry Command Line Interface \[page 378\]](#)

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 \[page 338\]](#)

### 3.1.16.4.1 Bind Service Instances to Applications Using the Cockpit

You can bind service instances to applications both at the application view, and at the service-instance view in the cockpit.

#### Prerequisites

- 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 \[page 62\]](#).
- Create a service instance. For more information, see [Create Service Instances Using the Cockpit \[page 372\]](#).

#### Procedure

1. Navigate to the space in which you deployed your application and created the service instance. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).

##### i Note

We have designed new cockpit screens to view and manage service instances and service bindings.

For more information, see [Binding Service Instances to Cloud Foundry Applications](#).

2. Choose one of the following options:

##### To create the binding at the view

of the

Do the following:

---

Application

1. In the navigation area, choose [Applications](#), then select the relevant application.
  2. In the navigation area, choose [Service Bindings](#).
  3. Choose [Bind Service](#).
  4. Choose a service type, then choose [Next](#).
  5. Choose a service, then choose [Next](#).
  6. To create a new instance of the service, choose [Create new instance](#) and follow the steps required for creating a new instance. To reuse an existing instance, choose [Re-use existing instance](#). Then choose [Next](#).
  7. Choose [Finish](#) to save your changes.
-

To create the binding at the view of the	Do the following:
Service instance	<ol style="list-style-type: none"> <li>In the navigation area, choose  Services &gt; Service Instances.</li> <li>Choose  (Bind Instance) in the Actions column for your service instance.</li> <li>Select your application.</li> <li>(Optional) Specify parameters in the JSON format or select a JSON file.</li> <li>Save your changes.</li> </ol>

### 3.1.16.4.2 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- 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 \[page 62\]](#).
- Create a service instance. For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 373\]](#).

#### 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 in a valid JSON object.

#### Related Information

[Creating Service Instances \[page 372\]](#)

[Creating Service Keys \[page 379\]](#)

[About Services \[page 371\]](#)

## 3.1.16.5 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 Cloud Platform cockpit or the Cloud Foundry Command Line Interface to create service keys:

- [Create Service Keys Using the Cockpit \[page 379\]](#)
- [Create Service Keys Using the Cloud Foundry Command Line Interface \[page 380\]](#)

For more information on service keys, see <https://docs.cloudfoundry.org/devguide/services/service-keys.html>

### 3.1.16.5.1 Create Service Keys Using the Cockpit

Use the cockpit to create a service key.

#### Prerequisites

Create a service instance. For more information, see [Create Service Instances Using the Cockpit \[page 372\]](#).

#### Procedure

1. Navigate to the space in which you've created a service instance for which you want to create a service key.

##### i Note

We have designed new cockpit screens to view and manage service instances, service bindings, and service keys in Cloud Foundry.

For more information, see [Creating Service Keys in Cloud Foundry](#).

2. In the navigation area, choose [Services](#) [Service Marketplace](#).

You see a list of all services that are available to you.

3. Choose the service for which you want to create a service key.
4. In the navigation area, choose [Instances](#), then select the instance you're creating a key for.

5. In the navigation area, choose *Service Keys*.
6. Choose *Create Service Key*.
7. Enter a name for the service key. Optionally enter configuration parameters.
8. Save your changes.

## Results

Local clients, apps in other spaces, or entities outside your deployment can now access your service instance with this key.

### 3.1.16.5.2 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- Create a service instance. For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface \[page 373\]](#).

#### 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 in a valid JSON object.

## 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 \[page 372\]](#)

[Binding Service Instances to Applications \[page 376\]](#)

[About Services \[page 371\]](#)

### 3.1.16.6 Deleting Service Instances

Use the SAP Cloud Platform 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 \[page 381\]](#)
- [Delete Service Instances Using the Cloud Foundry Command Line Interface \[page 382\]](#)

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 \[page 938\]](#)

#### 3.1.16.6.1 Delete Service Instances Using the Cockpit

You can use the cockpit to delete service instances.

## Procedure

1. Navigate to the space in which you want to delete a service instance.

#### ℹ Note

We have designed new cockpit screens to view and manage service instances.

For more information, see [Deleting Service Instances](#).

2. In the navigation area, choose Services Service Instances.

All services instances available in this space appear.

3. Choose the service instance which you want to delete.
4. Select (delete)
5. Confirm that you want to delete this service instance.

If your service instance is bound to an application, this step also removes the binding.

### 3.1.16.6.2 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

#### Procedure

1. Open a command line and log in.

```
cf l -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 \[page 371\]](#)

[Deleting Service Instances \[page 381\]](#)

### 3.1.16.7 Updating Service Instances

Use the Cloud Foundry Command Line Interface to update service instances:

- [Update Service Instances Using the SAP Cloud Platform Cockpit or Cloud Foundry Command Line Interface \[page 383\]](#)

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.

#### 3.1.16.7.1 Update Service Instances Using the SAP Cloud Platform Cockpit or Cloud Foundry Command Line Interface

Use the SAP Cloud Platform cockpit or the Cloud Foundry Command Line Interface to update service instances:

## Prerequisites

### i Note

For more information about how to use the SAP Cloud Platform cockpit to update Cloud Foundry service instances, see [Deleting Service Instances](#).

- 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

## 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`.

## 3.1.16.8 Boosters

Boosters are a set of guided interactive steps that enable you to select, configure, and consume services on SAP Cloud Platform to achieve a specific technical goal.

You can access them directly from your desired global account in the SAP Cloud Platform 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 \[Feature Set A\] \[page 762\]](#)

[Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

[Create Spaces \[page 922\]](#)

## 3.2 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](#).

With your ABAP applications, you can **consume**:

- HTTP services (HTTP client). See [HTTP Communication \[page 473\]](#).
- OData services (service consumption model). See [Developing a UI Service with Access to a Remote Service](#).
- Remote Function Calls (RFC) via Cloud Connector. See [Integrating On-Premise Systems \[page 478\]](#).
- External services. See [Set Up the Destination Service \[page 436\]](#).

### Development Resources

[ABAP RESTful Application Programming Model](#)

[ABAP Development User Guide](#)

[ABAP CDS Development User Guide](#)  
[ABAP for SAP HANA Development User Guide](#)  
[ABAP Keyword Documentation \[page 387\]](#)  
[ABAP Lifecycle Management \[page 387\]](#)  
[Connect to the ABAP System \[page 433\]](#)  
[HTTP Service Development \[page 476\]](#)  
[HTTP Communication \[page 473\]](#)  
[Integrating On-Premise Systems \[page 478\]](#)  
[Consuming External Services \[page 436\]](#)  
[Working with abapGit \[page 389\]](#)

## Related Information

[Eclipse Tool for the ABAP Environment](#)  
[Learning Journey](#)  
[Manage Software Components \[page 1071\]](#)  
[Tutorials!\[\]\(bbe046f0503ae09bc60ffc8919e4a1e0\_img.jpg\)](#)  
[Video Tutorials!\[\]\(907b86b73c1ba340aaf3cd2d7a5d39de\_img.jpg\)](#)

### 3.2.1 ABAP Development User Guides

Get an overview of the ABAP development user guides.

#### [ABAP Development User Guide](#)

This guide describes the functionality and usage of the possibilities within the ABAP Development Tools (ADT). It focuses on use cases for creating, editing, testing, debugging, and profiling development objects.

#### [ABAP CDS 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.

#### [ABAP RESTful Application Programming Model](#)

The ABAP RESTful Application Programming model supports the development of all types of SAP Fiori applications as well as publishing Web APIs.

- [Developing Read-Only List Reporting Apps](#)
- [Developing Unmanaged Transactional Apps Based on Existing Application Logic](#)
- [Developing a Web API](#)
- [Developing a UI Service with Access to a Remote Service](#)

#### [ABAP for SAP HANA Development User Guide](#)

This guide describes the basic idea to manage SAP HANA procedures and their lifecycle inside the ABAP server. To allow native consumption of SAP HANA features from within the ABAP layer, the SAP HANA database procedure language SQLScript has been integrated into the ABAP stack.

## 3.2.2 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](#).

## 3.2.3 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.

### 3.2.3.1 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.

#### i 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 cannot move development objects from one software component to another. Thus, the introduction of software components should be planned carefully
- 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.

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.

#### i Note

Objects assigned to this software component cannot be transported or moved to your custom software component.

- 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 of type Business Configuration. For details on how to work with business configuration data, see [Business Configuration for SAP Cloud Platform ABAP Environment](#).

### 3.2.3.2 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 Cloud Platform 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 \[page 389\]](#).

For a quick start on transportation, see [Transport a Software Component Between Two ABAP Systems](#).

#### 3.2.3.2.1 Working with abapGit

abapGit is an open-source Git client for ABAP. In the ABAP environment, it is used to import existing code into your cloud system.

Learn how to:

- [Install and Set Up abapGit \[page 390\]](#)
- [Create Content in an On-Premise System and Push it to abapGit Repository \[page 391\]](#)
- [Import Content from abapGit Repository into the ABAP Environment \[page 392\]](#)

#### Related Information

[Tutorial: Use abapGit to Transform ABAP Source Code to the Cloud](#)

[Released ABAP Object Types \[page 398\]](#)

### 3.2.3.2.1.1 Install and Set Up abapGit

Learn how to install and set up abapGit.

#### Prerequisites

- You have signed up for a Git account of your choice, for example GitHub.
- You have access to an on-premise system with the required root CA of the Git server (STRUST).
- You have downloaded and installed the front-end components of [ABAP Development Tools \(ADT\)](#). See [Video Tutorial: Configure Developer Tools](#).

#### ! 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. 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](#).
- Your repository is set up.
4. 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/larshp/abapGit>.

#### ⚠ Caution

Check with your system administrator before installing zabapgit.

5. Log on to an on-premise system of your choice, create a new program, and paste the content from your clipboard.

#### ℹ Note

Select EN as your logon language.

6. Activate and execute the program.  
abapGit is installed and launched. See also [Video Tutorial: abapGit Installation](#).
7. Log on to ABAP Development Tools in Eclipse.
8. Navigate to [Help > Install New Software...](#).
9. To install the abapGit repositories ADT plug-in, add the following URL: <http://eclipse.abapgit.org/updateSite/>.

10. To display all the available features, press enter, and select *abapGit for ABAP Development Tools (ADT)*.
11. Select *Next* to finish the installation.

## Next Steps

[Create Content in an On-Premise System and Push it to abapGit Repository \[page 391\]](#)

### 3.2.3.2.1.2 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 \[page 390\]](#).

##### **! 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*.
  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 \[page 398\]](#).
  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*.
  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 \[page 392\]](#)

### 3.2.3.2.1.3 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 \[page 391\]](#).
- You have access to an ABAP cloud system. See [Create an ABAP System \[page 48\]](#).
- You have defined a developer role and assigned a developer user in the ABAP environment. See [Define a Developer Role \[page 49\]](#) and [Assigning the ABAP Developer User to the ABAP Developer Role](#).

#### ! 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 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*.

#### i 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 \[page 398\]](#).

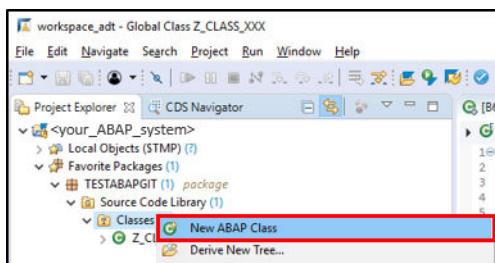
#### i Note

The number of imported objects can differ from the number of exported objects because only released ABAP object types are considered during the import.

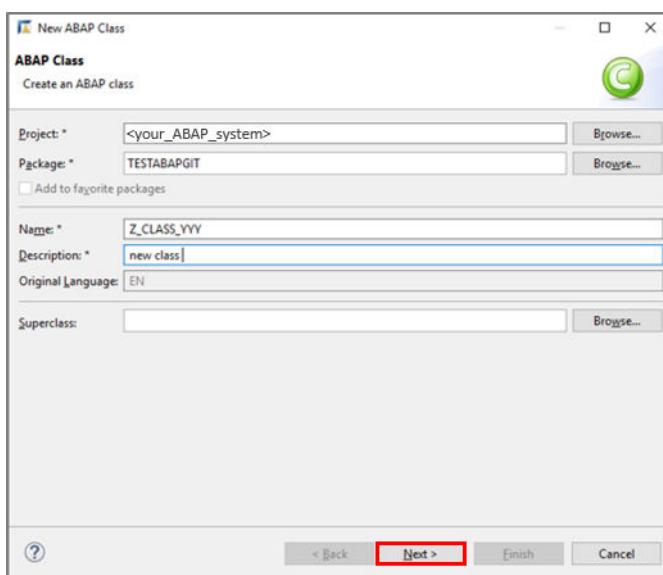
### 3.2.3.2.1.4 Transfer your ABAP Source Code via abapGit

Use abapGit to transfer your ABAP source code from an SAP Cloud Platform ABAP Environment instance back to your repository.

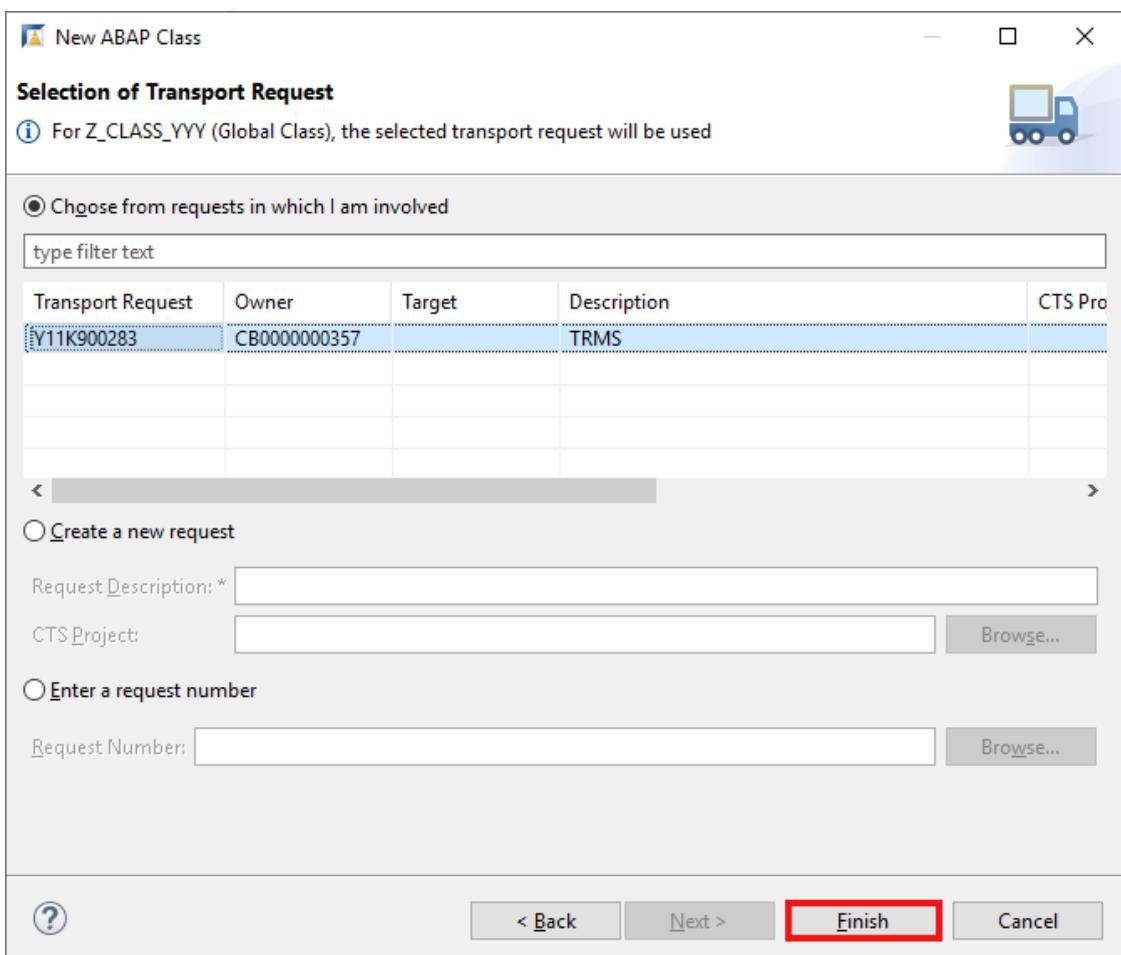
1. Create or adapt your source code in the SAP Cloud Platform ABAP Environment instance:
  1. Open **ABAP Development Tools** and logon to your ABAP system.
  2. Add an ABAP development object, e.g. an ABAP class, to your already existing *TESTABAPGIT* package. To do this, click on **Classes** and select **New ABAP Class**.



3. Enter a name and description for your new ABAP class and click **Next>..**



4. Click **Finish**.



- Now you can implement your newly created class.

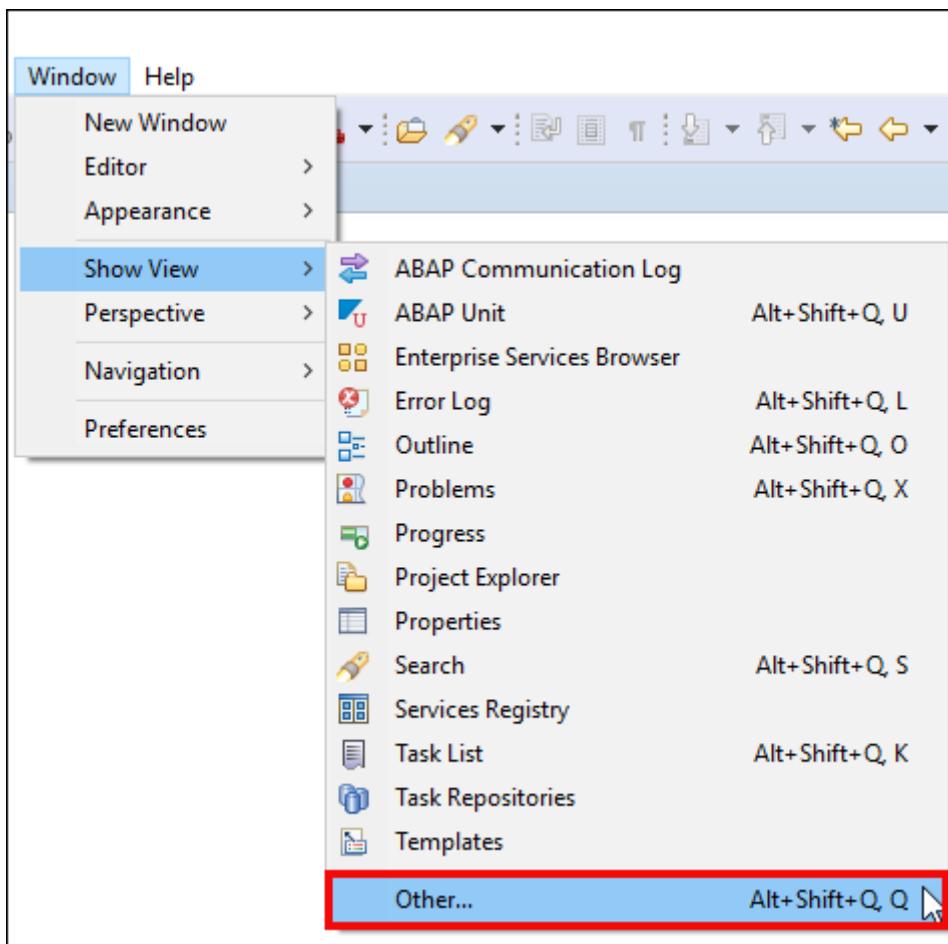
```

C [Y11] Z_CLASS_YYY ✎ C [Y11] Z_CLASS_XXX
▶ C Z_CLASS_YYY ▶
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.
13②  METHOD IF_OO_ADT_CLASSRUN~MAIN.
14    out->write('Hello world!').
15  ENDMETHOD.
16  ENDCLASS.|
```

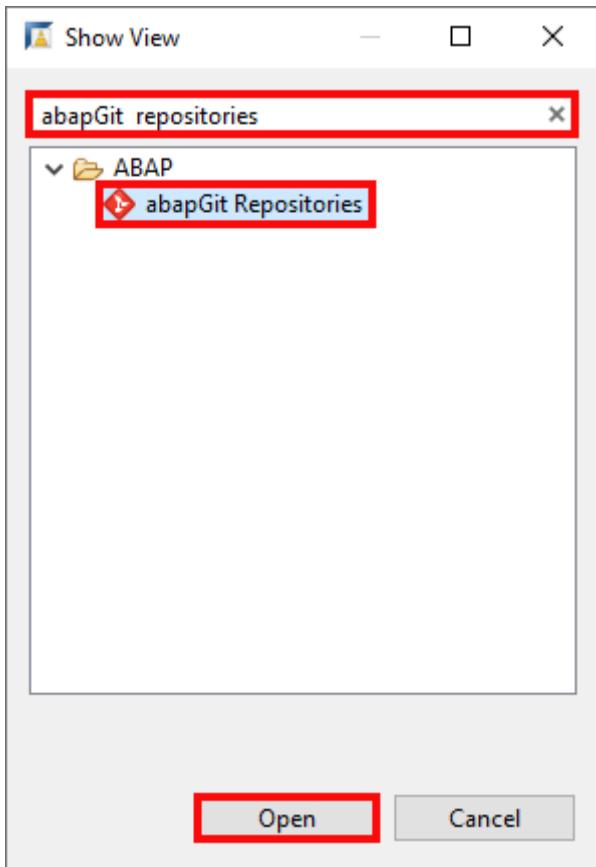
- Save and activate.

2. Open abapGit repositories:

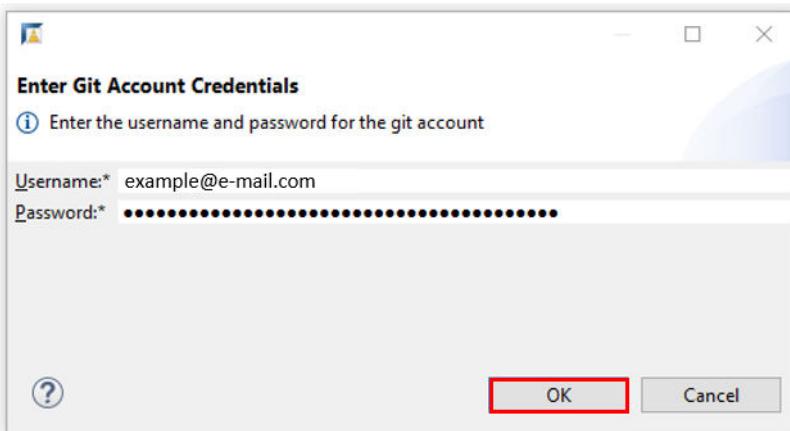
1. Select your **ABAP system** in the Project Explorer and select **Windows > Show View > Other** to open the *abapGit* repositories.



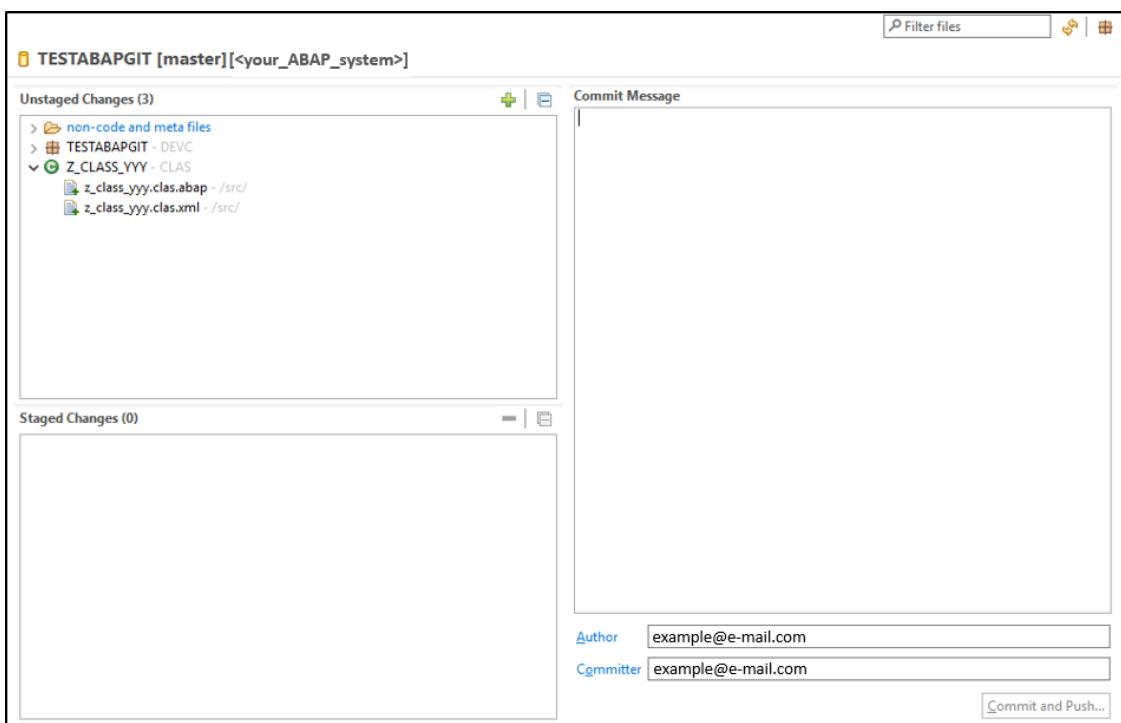
2. Search for *abapGit repositories*, select it and click **Open**.



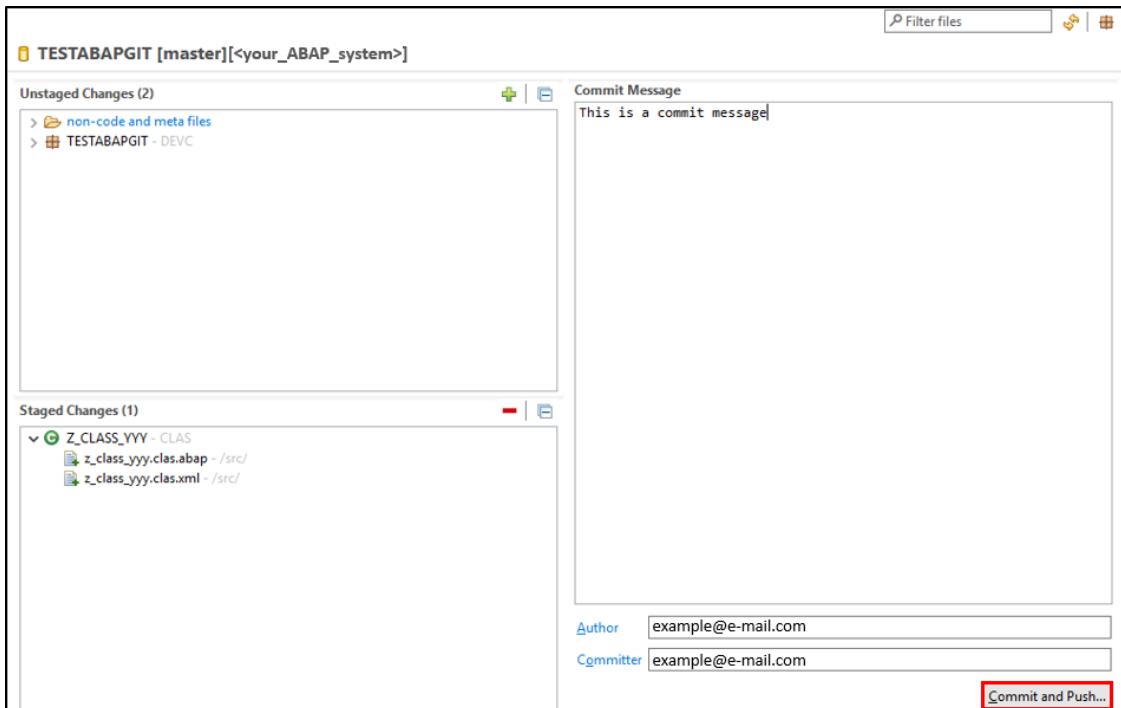
3. Stage and commit your ABAP development objects.
  1. Right-click your **TESTABAPGIT** package and click **Stage and Push**.
  2. Enter your repository credentials in the popup and click **OK**.



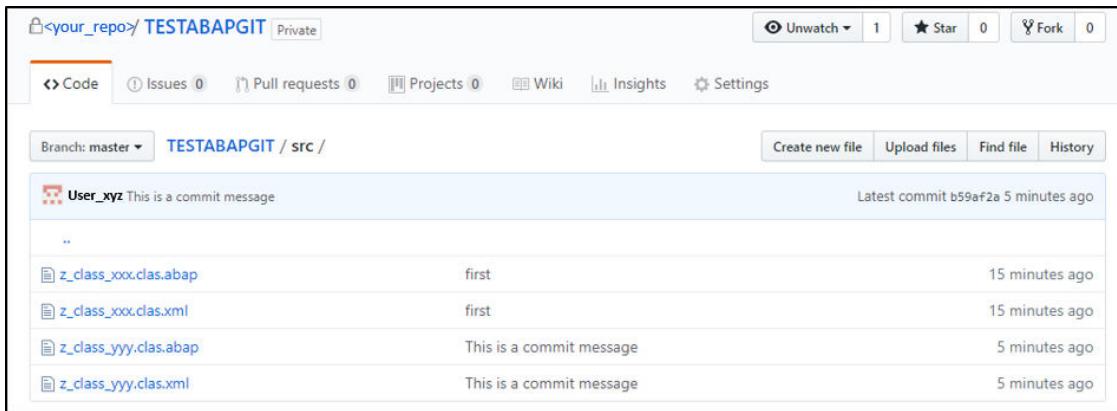
3. In the staging view you will see a list of changed, created or deleted ABAP development objects which can be transferred to your repository. The icon of the respective file indicates whether a file was created, modified or deleted.



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.



### 3.2.3.2.1.5 Released ABAP Object Types

The following table contains all the released ABAP object types that can be imported into your ABAP environment tenant.

APIS	API Release State of Objects
AUTH	Authorization Check Fields
BDEF	Behavior Definition
CLAS	Class
DCLS	ABAP Data Control Language Sources
DDLS	Data Definition Language Source
DDLX	CDS Metadata Extensions
DEVC	Package
DOMA	Domain
DTEL	Data Element
ENQU	Lock Object
ENHO	Enhancement Implementation Object
ENHS	Enhancement Spot Implementation Object
FUGR	Function Group
FUNC	Function Module
INTF	Interface
MSAG	Message Class
SRVB	Service Binding
SRVD	Service Definition
SUSO	Authorization Object
TABL	Table Definition

TTYP	Table Type
XSLT	Transformation

---

## Limitations

- Every import is imported as inactive.
- Pull overwrites all local changes.

## Known Issues

- Changes in table definitions could lead to data loss.

### 3.2.3.3 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. 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 Cloud Platform offers a service called ABAP System (abap). If you want to develop with ABAP in SAP Cloud Platform, you have to create an instance of that service, see [Create an ABAP System \[page 48\]](#). 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 master 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.

### i 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.

### 3.2.3.4 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 \[page 418\]](#)) 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.

### **3.2.3.5 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.

#### **3.2.3.5.1 Required Tools**

Different actions require different tools. In the process descriptions, the following tools are used:

- SAP Cloud Platform Cockpit (web-based administration interface)
- [Maintain Business Roles \[page 1043\]](#) (SAP Fiori app)
- [Maintain Business Users \[page 1038\]](#) (SAP Fiori app)
- [Manage Software Components \[page 1071\]](#) (SAP Fiori app)
- Eclipse Tool for the ABAP environment is an Eclipse installation with our 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 (e.g. for OData tests)
- External documentation tool to document test results and release decision if required

#### **3.2.3.5.2 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 Cloud Platform Cockpit
  - Needs authorization for space/organization management in the Cloud Foundry 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	Creation and maintenance of business users with Manage Business Users app
SAP_CORE_BC_IAM_RM	Identity and Access Management – Role Management	Maintain business roles with Manage Business Roles 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_PC	Lifecycle Management - Software Components	Software component creation and import with Manage Software Components app
SAP_A4C_BC_TRN_REL_PC	Development - Transport Release Management	Releasing transport requests in Eclipse with ADT
SAP_CORE_BC_IAM_UM	Identity and Access Management - User Management	Un-/lock developers for corrections in correction ABAP systems with Manage Business Users app
SAP_CORE_BC_IAM_RA	Identity and Access Management – Role Assignment	En-/disable developers for corrections in correction ABAP Systems by role assignment with Manage Business Users app

- Tester
  - Tests the software

#### i 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.

### **3.2.3.5.3 Important Notes**

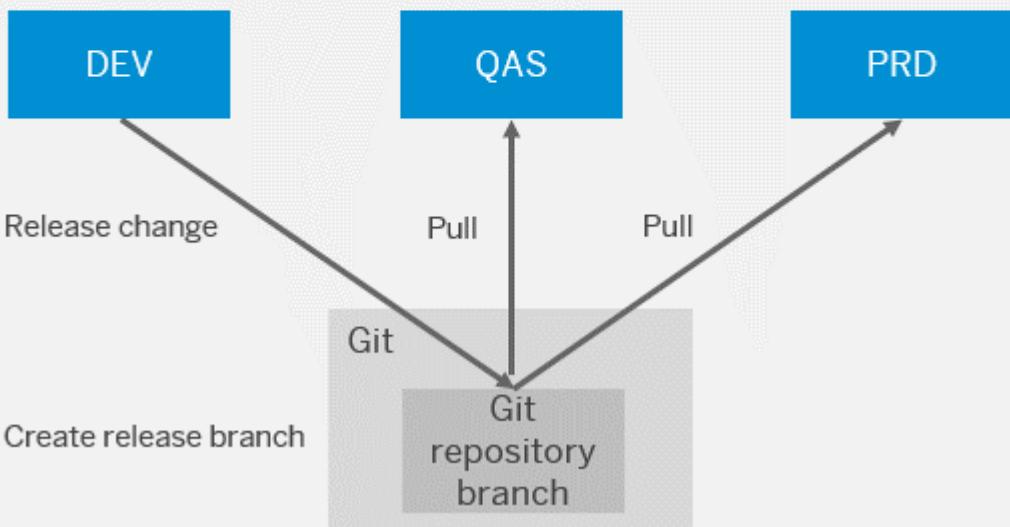
- The creation of a Git repository branch creates a clone of the parent branch. If you create a branch immediately after the release decision, you have a “frozen” state. You can import this state safely into a production ABAP system or multiple production ABAP systems, “pre-production” ABAP systems, etc. or you may use it later for auditing.
- In the different system setups, the ABAP systems can either be provisioned at once upon development start or just in time 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 ABAP system.
- A classical 3-system setup comprising a development, quality assurance, and production ABAP system can be enhanced by pre-production ABAP systems, multiple test ABAP systems, etc. Test ABAP systems can be provisioned on demand (temporary) or permanently.

### **3.2.3.5.4 Use Case 1: One Codeline in a 3-ABAP-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 ABAP system.

## SAP Cloud Platform Global Customer Account



## For Go Live/Development After Go Live (Including Deferrable Corrections)

### Starting situation for Go Live

Recently created development ABAP 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 ABAP 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 ABAP system DEV is based on the master branch
- Quality Assurance ABAP system QAS and production ABAP 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

The QAS ABAP system has always the same software state as the PRD ABAP system, unless a new change is tested and released. This means, transport requests are released in DEV ABAP systems only if development is completed and it is planned to import the changes to the production ABAP 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	At Go Live only: Create the software component and pull it initially	Manage Software Components app
1	DEV	Developer	Develop new functionality or a deferrable correction. All changes are collected in transport requests	ADT for Eclipse
2	DEV	Release Manager	Release the transport request. The changes are now in the master branch	ADT for Eclipse: Transport Organizer
3	QAS		Check out master branch/at Go Live: pull software component into QAS	Manage Software Components app
4	QAS		Test the change and report test result	ADT for Eclipse and custom SAP Fiori apps & External test tools  External documentation tool
			If changes are required, repeat steps 1-4	
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> (at Go Live: YYYY-01)	Manage Software Components app
7	QAS	Release Manager	Check out the new release branch YYYY-<nn+1> (@Go Live: YYYY-01) into QAS	Manage Software Components app
8	PRD	Release Manager	Check out the new release branch YYYY-<nn+1> (at Go Live: YYYY-01) into PRD	Manage Software Components app

## Urgent Corrections

- Development ABAP system DEV is based on the master branch
- Quality assurance ABAP system QAS and production ABAP 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

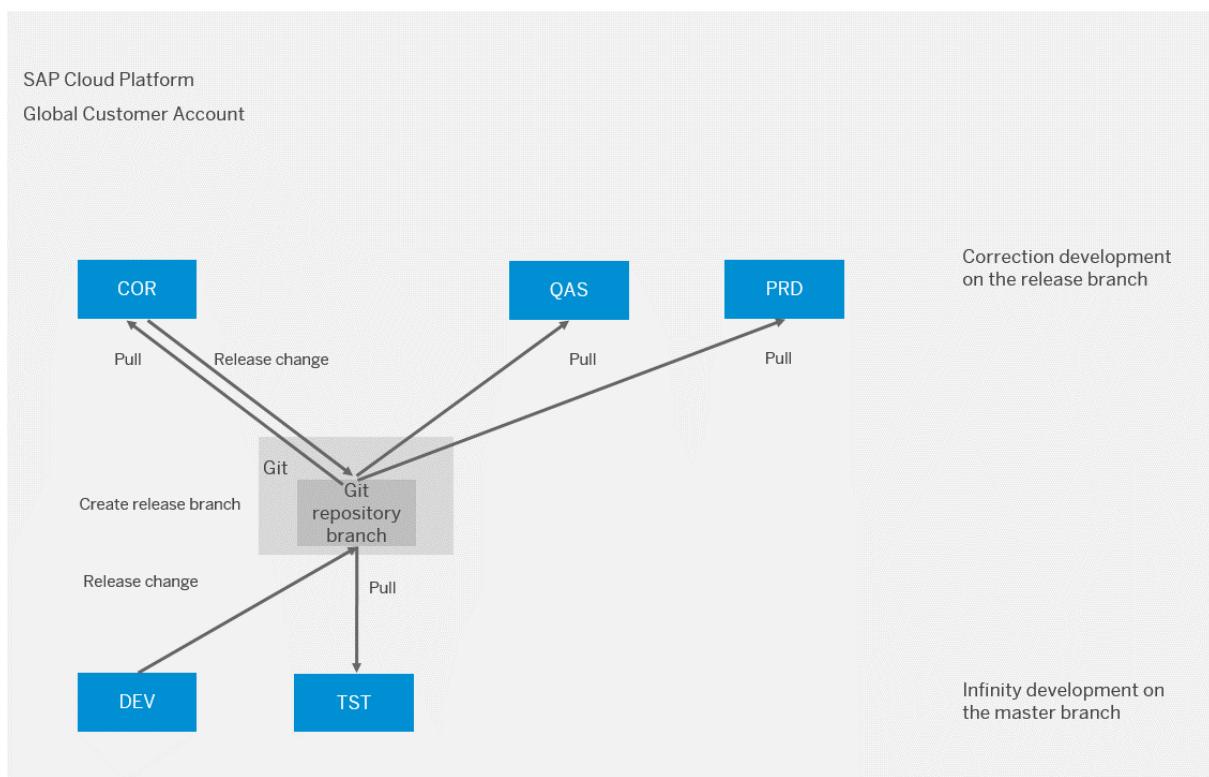
This process differs from the previous one in the branch it is developed in. As the correction is too urgent to release it with the next development release only, it is done in the release branch. To achieve this separation, all current development activities need to be paused because the DEV ABAP system needs to check out the latest release branch instead of the master branch.

Step	System	Role	Task	Tool
1	DEV	Release Manager	Check out the release branch YYYY-<nn>	Manage Software Components app
2	DEV	Developer	Fix existing functionality. All changes are collected in transport requests	ADT for Eclipse
3	DEV	Release Manager	Release the transport request. The changes are now in the master branch	ADT: Transport Organizer
4	QAS	Release Manager	Pull the software component to get correction into already checked out release branch YYYY-<nn>	Manage Software Components app
5	QAS	Tester	Test the change and report the test result	ADT 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 to get correction into already checked out release branch YYYY-<nn>	Manage Software Components app
8	DEV	Release Manager	Check out the master branch in DEV	Manage Software Components app

Step	System	Role	Task	Tool
9	DEV	Developer	Perform the same changes as for the correction in the master branch and release them	ADT for Eclipse

### 3.2.3.5.5 Use Case 2: One Development and Correction Codeline in a 5-ABAP-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 ABAP system before being delivered to production.



General considerations:

- ABAP systems COR and QAS have the same software state as the production ABAP system PRD, unless a new change is tested and released. This means transport requests are released in the DEV ABAP system only if development is completed and it is planned to import the changes to the production ABAP system.
- Releases are planned and communicated to development in advance:
  - 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 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 ABAP system(s) PRD
- ABAP 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 the correction ABAP system COR.

## For Go Live/Development after Go Live (Including Deferrable Corrections)

### Starting Situation for Go Live:

Recently created development ABAP 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 ABAP 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 ABAP system DEV and test ABAP system TST are on the master branch
- Correction ABAP system COR, quality assurance ABAP system QAS, and production ABAP 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 and pull it initially	Manage Software Components app
1	DEV	Developer	Develop a new functionality or a deferrable correction. All changes are collected in transport requests	ADT for Eclipse
2	DEV	Developer	Once development is finished, release the transport request. The changes are now in the master branch	ADT for Eclipse: Transport Organizer view
3	TST	Release Manager	Pull the software component into system TST	Manage Software Components app
4	TST	Tester	Test the change and report the test result	ADT for Eclipse and custom SAP Fiori apps as well as external test tools  External documentation tool
			If changes are required, repeat steps 1-4	
5	QAS or any other	Release Manager	Cutoff: at cutoff date, create a release branch YYYY-<nn+1> (at Go Live: YYYY-01) (release candidate)	Manage Software Components app
6	QAS	Release Manager	Check out the release branch YYYY-<nn+1> (at Go Live: YYYY-01) into system QAS	Manage Software Components app
7	QAS	Tester	Test the release candidate and report the test result	ADT 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> (at Go Live: YYYY-01) into system COR	Manage Software Components app

Step	System	Role	Task	Tool
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	Manage Business Users app
10	COR	Developer	Implement the correction	ADT for Eclipse
11	COR	Release Manager	Release the transport request. The changes are now in the release candidate.	ADT for Eclipse: Transport Organizer
12	QAS	Release Manager	Pull the software component to get the correction into the already checked out release branch YYYY-<nn+1> (@Go Live: YYYY-01)	Manage Software Components app
13	QAS	Tester	Test the change and report the test result	ADT 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> (at GoLive: YYYY-01) into system PRD	Manage Software Components app
16	COR	Release Manager	Disable the respective development users for development in system COR	Manage Business Users app
17	DEV	Developer	Perform the same changes as for the correction in the master branch and release them	ADT for Eclipse

Step	System	Role	Task	Tool
18	TST	Release Manager	Pull the software component to get the correction into the already checked out master branch	Manage Software Components app

## 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 master branch of the development ABAP system, release them, and create the new release branch YYYY-<nn+2> derived from that master. 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.

### i 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 ABAP system DEV and test ABAP system TST are on the master branch
- Correction ABAP system COR, quality assurance ABAP system QAS, and production ABAP 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.  Create a transport request.	Manage Business Users app

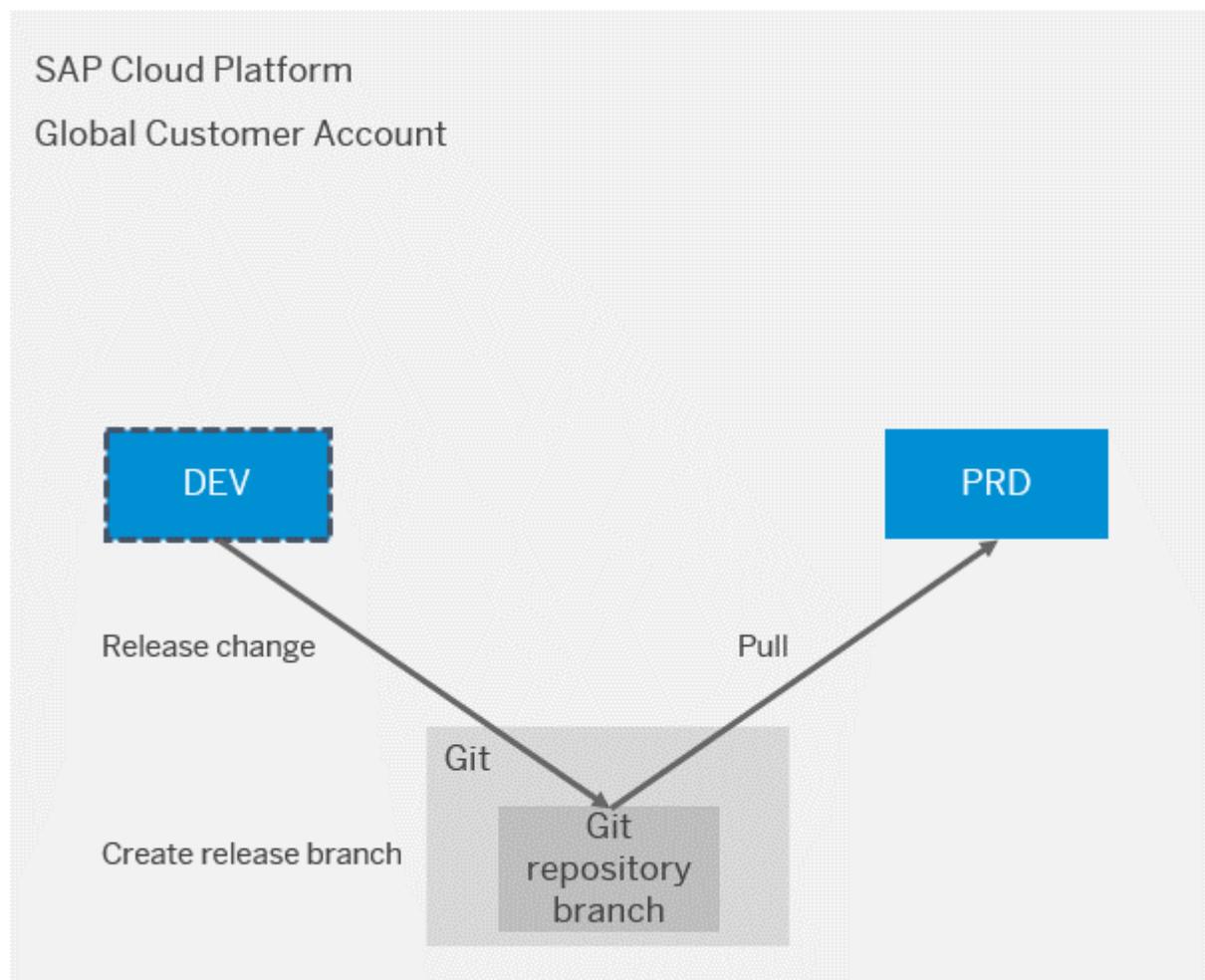
Step	System	Role	Task	Tool
2	COR	Developer	Implement the correction	ADT for Eclipse
3	COR	Release Manager	Release the transport request. The changes are now in the release candidate	ADT for Eclipse: Transport Organizer
4	QAS	Release Manager	Pull the already checked out branch YYYY-<nn+1> into QAS	Manage Software Components app or external tool calling the pull service of communication scenario Test Integration
5	QAS	Tester	Test the change and report the test result	ADT 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	As the correction was successfully tested before it gets approved now.	External documentation tool
7	PRD	Release Manager	Check out the release branch YYYY-<nn+1> into system PRD	Manage Software Components app
8	COR	Release Manager	Disable the respective development users for development in system COR	Manage Business Users app
9	DEV	Developer	Perform the same changes as for the correction in the master branch and release them	ADT for Eclipse
10	TST	Release Manager	Pull the software component to get correction into already checked out master branch	Manage Software Components app

### 3.2.3.5.6 Use Case 3: One Codeline with On-Demand Development ABAP systems

You can apply this setup if you have:

- Small development projects (e.g. one SAP Fiori application, one developer)
- Occasional development activities (e.g. 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 ABAP system and an on-demand development ABAP system.



The advantage of this setup is that you have to administer few ABAP systems and only pay for the development ABAP system during the development periods. The payment model according to the lifetime of a system requires a SAP Cloud Platform enterprise agreement contract.

The disadvantages of this setup are that de-commissioning the development ABAP 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 takes the risk of forgetting to transport an object. This risk might be minimized a little 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 \[page 402\]](#).

We thus strongly recommend developing with [Use Case 1: One Codeline in a 3-ABAP-System Landscape \[page 404\]](#) instead of use case 3. The quality assurance ABAP system might be de-/commissioned on demand then.

## For the Go Live

### Starting situation:

- A (temporary) development ABAP system is set up
- The production ABAP system is set up
- Both ABAP systems are based on the master branch

Step	System	Role	Task	Tool
1	DEV, PRD	System Administrator	Provision a development and a production ABAP system	SAP Cloud Platform Cockpit
2	DEV, PRD	User Administrator	Create users and maintain roles	Manage Business Users and Manage Business Roles app
3	DEV	Release Manager	Create a software component	Manage Software Components app
4	PRD	Release Manager	Pull (empty) software component	Manage Software Components app
5	DEV	Developer	Develop new functional requirements or correct existing functionalities. All required changes are collected in transport requests	Eclipse for ADT
6	DEV	Tester	Test the change and report the test results	ADT 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	

Step	System	Role	Task	Tool
7	DEV	Release Manager	Release decision: the changes are successfully tested and approved.	External documentation tool
8	DEV	Release Manager	Release the transport request	ADT for Eclipse: Transport Organizer view
9	DEV	Release Manager	Create a release branch YYYY-<nn>	Manage Software Components app
10	PRD	Release Manager	Check out the release branch YYYY-<nn> of the software component	Manage Software Components app
11	DEV	System Administrator	De-provision ABAP system DEV	SAP Cloud Platform Cockpit

## Productive Usage

- The production ABAP system is based on the master 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 master 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 ABAP system	SAP Cloud Platform Cockpit
2	DEV	Release Manager	Create users, maintain and assign roles	Manage Business Users and Manage Business Roles app
3	DEV	Release Manager	Import master branch of software component	Manage Software Components app
4	DEV	Developer	Develop features	ADT for Eclipse
5	DEV	Developer	Release all transport requests	ADT for Eclipse

Step	System	Role	Task	Tool
6	DEV	Tester	Test new developments/fixes  If changes are required, repeat steps 3-5	ADT for Eclipse and custom SAP Fiori apps as well as external test tools
7	PRD	Release Manager	Pull the master branch into the PRD ABAP system	Manage Software Components app
8	DEV	Release Manager	Delete the development ABAP system	SAP Cloud Platform Cockpit

### 3.2.3.5.7 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 ABAP system.

As there is no selective picking of transport requests to merge corrections back into the master branch, this is manual effort.

You can use the *Compare Editor* in ADT to merge corrections back to the master branch.

The process would be as follows:

You can compare the latest version of an object in your ADT project in a hotfix ABAP system with the latest version of the same object in the development ABAP system. If there are new 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 your ADT.

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.	ADT for Eclipse: Transport Organizer View

Step	System	Role	Task	Tool
2	COR, DEV	Developer	<ol style="list-style-type: none"> <li>1. Select the resources in the Project Explorer in the correction ABAP system COR</li> <li>2. From the resource's pop-up menu, select Compare With</li> <li>3. Select the ABAP project that points to your development ABAP system DEV</li> </ol>	ADT for Eclipse: Project Explorer
3	DEV	Developer	Merge the changes into the ABAP system DEV	ADT for Eclipse

### 3.2.3.6 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, the complete content of the Git repository is 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

### 3.2.3.7 Automate the Software Lifecycle Management Process

To facilitate your software lifecycle management process, you can automate it by using communication scenario SAP\_COM\_0510. See [Pulling Git Repositories to an ABAP Environment System \[page 422\]](#).

### 3.2.3.8 Test Integration (SAP\_COM\_0510)

The Communication Scenario SAP\_COM\_0510 enables you to create Continuous Integration pipelines for SAP Cloud Platform ABAP Environment systems. With the APIs belonging to this Communication Scenario, you can import your ABAP code to the SAP Cloud Platform ABAP Environment system and check your code with the ABAP Test Cockpit (ATC).

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 to reuse. There, you can find various steps that implement the functionality of the APIs of SAP\_COM\_0510.

Additionally, there is an example pipeline for a Continuous Integration process that covers the following:

- creating a SAP Cloud Platform ABAP Environment system
- setting up a Communication Arrangement for the APIs
- importing a software component / Git repository
- running ATC checks
- deprovisioning the SAP Cloud Platform ABAP Environment system

Please find the documentation regarding project "Piper" here: <https://sap.github.io/jenkins-library/>.

#### 3.2.3.8.1 API to Manage Git Repositories

You can use the `MANAGE_GIT_REPOSITORY` API to pull 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.

##### i 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.

#### Related Information

[Pulling Git Repositories to an ABAP Environment System \[page 422\]](#)

[Working with Branches on a Git Repository \[page 425\]](#)

### 3.2.3.8.1.1 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 \[page 1020\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems \[page 1024\]](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 1022\]](#).
- 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.

##### i 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 master branch is automatically selected.

## 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": "master"
}
```

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='guid'02ceab7d-ff04-1eda-blea-eb7fdbf28bc4',sc_name='ZHM_INITIAL_TEST',branch_name='')",
                    "uri": "host.com/sap/opu/odata/sap/MANAGE_GIT_REPOSITORY/Clones(uuid='guid'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 ...
        ]
    }
}
```

## 3.2.3.8.1.2 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 \[page 1020\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems \[page 1024\]](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 1022\]](#).
- You have selected the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.

### Context

You can use the communication scenario **SAP\_COM\_0510** to pull Git repositories to an ABAP environment system.

#### i 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,
      }
    ]
  }
}
```

### 3.2.3.8.1.3 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 \[page 1020\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems \[page 1024\]](#).
- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 1022\]](#).
- 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 master branch. A master 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 Cloud Platform ABAP Environment system.

## i 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. **(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 \[page 422\]](#).
3. **(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" : "master",
  "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": "master",
    "created_by": "CC0000000001",
    "created_on": "/Date(1584967657000+0000)/",
    "last_commit_on": "/Date(1584634658000+0000)/"
}
}

```

4. **(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/
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": "master",
        "created_by": "CC0000000001",
        "created_on": "/Date(1584967657000+0000)/",
        "last_commit_on": "/Date(1584634658000+0000)/"
    }
}

```

#### i Note

Please note that the newly created branch is not yet active.

5. **(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 \[page 422\]](#)). This entity is specified in the \_\_metadata of the response body.

### 3.2.3.8.2 Executing ABAP Test Cockpit (ATC) Check Runs

#### Prerequisites

- You have an SAP Cloud Platform ABAP Environment system.
- You've created a *Communication User* as described in [How to Create Communication Users \[page 1020\]](#).
- You've created a *Communication System* as described in [How to Create Communication Systems \[page 1024\]](#).

- You've created a *Communication Arrangement* as described in [How to Create a Communication Arrangement \[page 1022\]](#).
  - You have selected the communication scenario **SAP\_COM\_0510** for your communication arrangement and have mapped it to your communication system.

## 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 will learn how to trigger an ATC Check Run via REST Service.

For more information about 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

## Headers

KEY	VALUE
accept	application/vnd.sap.atc.run.v1+xmlapplication/ 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

KEY	VALUE
x-csrf-token	<token>

KEY	VALUE
location	/sap/bc/adt/atc/runs/ 00000000000000000000000000000000

2. **(Trigger an ATC Check Run)** To trigger 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 which belong to the specified software components.

For the package value, enter the name of the package you want to have checked.

To include subpackages, you need to state `includeSubpackages="true"`.

```
<?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">
  <obj:objectSet>
    <obj:softwarecomponents>
      <obj:softwarecomponent value="SAP_BASIS"/>
    </obj:softwarecomponents>
    <obj:packages>
      <obj:package value="Z_MY_PACKAGE" includeSubpackages="true"/>
    </obj:packages>
  </obj: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 will 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="/sap/bc/oo/classes/.../source/main">
        <error message="Hard-coded user name"
source="CL_CI_TEST_EXTENDED_CHECK_SEC#0821" line="2" severity="error"/>
    </file>
</checkstyle>
```

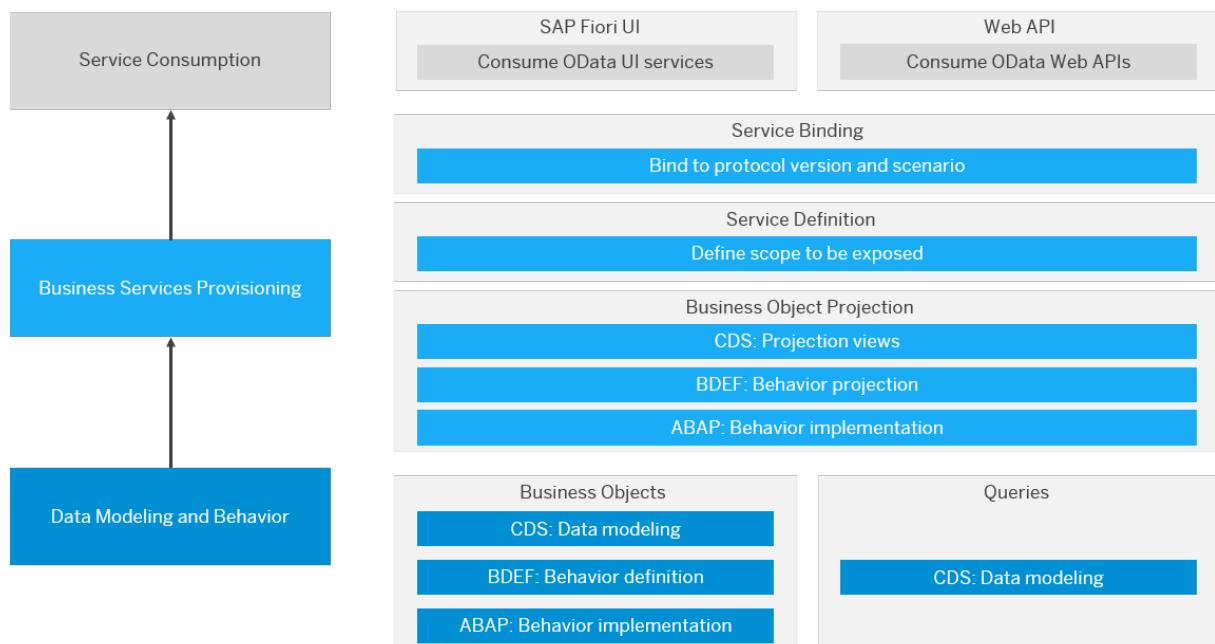
### 3.2.3.9 Software Assembly Integration (SAP\_COM\_0582)

The Software Assembly Integration Communication Scenario SAP\_COM\_0582 is not released for general usage. If you want to use it, please contact SAP.

### 3.2.4 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



- [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#loiof2cbcacaf8b74540b0708fc143875bc3\\_\\_Web\\_API](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiof2cbcacaf8b74540b0708fc143875bc3__Web_API) [[https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiof2cbcacaf8b74540b0708fc143875bc3\\_\\_Web\\_API](https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/f2cbcacaf8b74540b0708fc143875bc3.html#loiof2cbcacaf8b74540b0708fc143875bc3__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/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/511a134f89614e77a2231d0af5b924f8.html> [https://help.sap.com/viewer/923180ddb98240829d935862025004d6/Cloud/en-US/511a134f89614e77a2231d0af5b924f8.html]
- <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]

## 3.2.5 Connect to the ABAP System

Set up your ABAP cloud project to connect to the ABAP system.

### Prerequisites

- 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 \[Feature Set A\]](#) [page 923].
- You are assigned the developer role **or** you have a service key in JSON format. See [Define a Developer Role](#) [page 49], [Assigning the ABAP Developer User to the ABAP Developer Role](#), and [Creating Service Keys](#) [page 379].

## Procedure

1. Open Eclipse and select from the menu.  
A *Select a wizard* dialog box opens.
2. Open the *ABAP* folder, choose *ABAP Cloud Project*, and select *Next*.
3. 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 Cloud Platform ABAP Environment* page, select the *SAP Cloud Platform 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 Cloud Platform 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 Cloud Platform 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.
4. Select *Next*.
5. 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.
  - 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.

### 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.
6. 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

### i Note

Currently, English is the only logon language.

7. Select *Next*.

The *Project Name and Favorite Packages* page is opened.

8. [Optional:] If you want to change the name of the project, enter a new name in the *Project Name* field.

### i 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.

9. To create your ABAP cloud project, select *Finish*.

## Results

You have created an ABAP cloud project that is added to the *Project Explorer*.

### i 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 Console Application with SAP Cloud Platform ABAP Environment](#) 

[ABAP Cloud Projects](#)

## 3.2.6 Integration and Connectivity

### 3.2.6.1 Consuming External Services

Get more information about consuming external services.

Learn how to:

- Set Up the Destination Service [page 436]
- Use the Destination Service in Your ABAP Code with OAuth2 Client Credential Grant [page 437]
- Use the Destination Service in Your ABAP Code with OAuth 2.0 SAML Bearer Assertion Flow [page 438]

#### 3.2.6.1.1 Set Up the Destination Service

##### Prerequisites

###### i Note

The following procedure is optional. We recommend that you configure the destinations directly on subaccount level. See also [Integrating On-Premise Systems \[page 478\]](#).

- You have created an ABAP service instance. See [Creating an ABAP System](#).
- You have created a destination service instance in the same subaccount. See [Creating a Destination Service Instance \(Optional\)](#).
- You have created a service key. See [Creating a Service Key for the Destination Service Instance \(Optional\)](#).

##### Procedure

1. To set up the destination service in the SAP Cloud Platform Cockpit, navigate to [Global Accounts](#).
2. Select your global account and your subaccount.
3. In the menu, go to [Spaces](#) and select the space that contains the ABAP service instance.
4. Expand [Services](#) and select [Service Instances](#).
5. To open the administration launchpad, you have the following options:
  - a. In the [Actions](#) column, choose the [Open Dashboard](#) icon.  
The administration launchpad is opened.
  - b. In the [Name](#) column, choose your ABAP service instance and select the [Open Dashboard](#) button.  
The administration launchpad is opened.

6. If necessary, provide your logon credentials to access the administration launchpad.
7. In the *Communication Management* section, select the *Communication Arrangements* tile.
8. On the *Communication Arrangements* page, select *New*.
9. 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).
10. Enter the service key of your destination service instance and select *Create*.

## Results

You have set up the integration between your ABAP service instance and your destination service instance.

### 3.2.6.1.2 Use the Destination Service in Your ABAP Code with OAuth2 Client Credential Grant

#### Prerequisites

- [Optional] You have set up the communication arrangement for scenario SAP\_COM\_0276. See [Creating a Communication Arrangement for the Destination Service Instance in the ABAP Environment \(Optional\)](#).
- You have an existing destination in your destination service instance.
- You have created an HTTP service or OData service. See [Tutorial: Create an HTTP Service](#), [Creating an OData Service](#) and [Video Tutorial: Create OData Service](#).

#### Procedure

1. In Eclipse, navigate either to your HTTP service or your OData service implementation.
  2. Create a destination object using class `cl_http_destination_provider` and method `create_by_cloud_destination` with the following parameters:
    - **i\_service\_instance\_name:** the value of the service instance name property of the communication arrangement
    - **i\_name:** the name of the destination
    - **i\_authn\_mode:** `if_a4c_cp_service=>service_specific`
- i Note**

This parameter is used to call the destination service with OAuth2 client credential grant.
3. Create an HTTP client object with class `cl_web_http_client_manager` and method `create_by_http_destination` providing the destination object you have created as a parameter.

4. Execute the request using method `execute` on the HTTP client object.

### 3.2.6.1.3 Use the Destination Service in Your ABAP Code with OAuth 2.0 SAML Bearer Assertion Flow

#### Prerequisites

- [Optional] You have set up the communication arrangement for scenario SAP\_COM\_0276. See [Creating a Communication Arrangement for the Destination Service Instance in the ABAP Environment \(Optional\)](#).
- You have an existing destination in your destination service instance.
- You have created an HTTP service or OData service. See [Tutorial: Create an HTTP Service](#), [Creating an OData Service](#) and [Video Tutorial: Create OData Service](#).

#### Procedure

1. In Eclipse, navigate either to your HTTP service or your OData service implementation.
2. Create a destination object using class `cl_http_destination_provider` and method `create_by_cloud_destination` with the following parameters:
  - `i_service_instance_name`: the value of the service instance name property of the communication arrangement
  - `i_name`: the name of the destination
  - [Optional] `i_authn_mode`: `if_a4c_cp_service=>user_propagation`
3. Create an HTTP client object with class `cl_web_http_client_manager` and method `create_by_http_destination` providing the destination object you have created as a parameter.
4. Execute the request using method `execute` on the HTTP client object.

#### i Note

This parameter is used to call the destination service with OAuth 2.0 SAML Bearer Assertion Flow. By default, it is set to `user_propagation`.

## 3.2.6.2 Consumption Scenarios

### 3.2.6.2.1 Consuming a Web Service

Learn more about consuming synchronous SOAP-based web services for outbound communication.

#### Overview

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 (ADT), see [Creating Service Consumption Model](#).

For the consumption type web service, other than for example OData, you need to provide the WSDL of the service you want to consume via the local file system. See [Generating Proxies for Remote Web Service](#). Upon successful activation, all the dependent objects, such as the enterprise service, corresponding dictionary objects, and the generated ABAP class, are created and displayed in the ADT project explorer.

For any operation of the web service, a code snippet is displayed in ADT that indicates how to call the service via the respective method of the generated class. It is instantiated via a destination object containing information such as the endpoint URL and authentication method. You can either create such a destination object in your code or retrieve it from a destination service.

#### → Recommendation

We recommend retrieving your destination object from a destination service if a destination service is available for your system. For more information, see [SOAP Communication \[page 443\]](#).

The metadata of your service consumption model display the so-called enhanced Web Services Description Language (WSDL). It describes how schema/WSDL entities are mapped to ABAP, and are enhanced by their ABAP names and types in particular. You can change all of these ABAP properties by editing the enhanced WSDL and re-activating the service consumption model. To learn more about the enhanced WSDL, see [Enhanced Web Services Description Language \(WSDL\) \[page 440\]](#).

#### Feature Scope

- Synchronous outbound web service communication
- Self-contained WSDLs (no import statements)
- Destination configuration options for the following authentication methods:
  - No authentication
  - Basic authentication
- gCTS support
- Support for proxy type **Internet** and **OnPremise**

## What You Need to Know

### → Recommendation

We recommend creating a service consumption model in an empty package. Dependent enterprise service objects are reused between different service consumption models in the same package.

### i Note

You can view dictionary objects and the generated ABAP class in the project explorer for an inactive service consumption model, but you can only create them upon activation of the service consumption model.

## Comparison Between On Premise and Cloud

- The service consumption model takes over the role of the service consumer
- The creation of a service consumption model in ADT is very similar to the creation of a service consumer from an external WSDL within the web service wizard in SAP GUI
- The idea of persisted logical ports shifted towards a transient concept. The corresponding data, such as the endpoint URL and authentication method, now reside in a destination, that can be retrieved from a destination service
- The enhanced WSDL is similar to the Design Time WSDL in the Proxy Editor in SAP GUI enhanced by ABAP properties
- The enhanced WSDL replaces the Proxy Editor in SAP GUI

## Related Information

[Creating Service Consumption Model](#)

[SOAP Communication \[page 443\]](#)

[Enhanced Web Services Description Language \(WSDL\) \[page 440\]](#)

### 3.2.6.2.1.1 Enhanced Web Services Description Language (WSDL)

The enhanced Web Services Description Language (WSDL) is a WSDL enriched by ABAP information that are relevant for design time.

When you use a WSDL as an input to create a service consumption model, the information relevant for design time, such as the types, messages, and port type sections, are analyzed and interpreted by the ABAP properties. If needed, these ABAP properties can then be changed within the enhanced WSDL to adjust the ABAP artifacts accordingly (upon re-activation). The mapping between XSD and ABAP types in particular can be tailored this way. This results in an internal representation, where the ABAP name and type proposals are

derived from. The enhanced WSDL is built from this internal representation and includes ABAP properties. That means, it is enhanced by ABAP properties. For any service consumption model, you can consider the enhanced WSDL as its service metadata.

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 contains additional nodes within this namespace to accommodate its ABAP properties. If there is an `abap` prefix, you can adjust the ABAP properties. These additional nodes can be XSD attributes or elements depending on the location of the node you want to enhance. This is necessary to maintain WSDL validity. You can enhance schema nodes, such as

`xmlns:xsd=http://www.w3.org/2001/XMLSchema`, via additional XSD attributes. WSDL nodes can be enhanced via additional XSD (sub)elements if `xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/` applies. Despite the way they are encoded into the enhanced WSDL, regular ABAP properties of any node comprise the prefix, name, description, and 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. You can choose any of the listed types by copying and pasting it into the `type` attribute of the given element.

#### Summary of Additional Nodes and Their Technical Types

Node name	Technical Type	Description
<code>abap:prefix</code>	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)
<code>abap:name</code>	Up to 30 characters, case insensitive	ABAP name
<code>abap:type</code>	String, comma-separated list of 30 characters	Possible ABAP types (only for simple types)
<code>abap:availableTypes</code>	Maintain Business Roles	Only services whose originals reside in the current system
<code>abap:description</code>	Up to 60 characters	Description
<code>abap:prefixT</code>	See prefix	Prefix for corresponding table type (if <code>maxOccurs &gt; 1</code> and additional table type is generated and a prefix is required)
<code>abap:nameT</code>	See name	Name for corresponding table type (if <code>maxOccurs &gt; 1</code> and additional table type is generated)
<code>abap:descriptionT</code>	See description	Description for corresponding table type (if <code>maxOccurs &gt; 1</code> 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_REQE" 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_REQE` 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 `wsdl`) rather than attributes:

#### ↳ Sample Code

```

<wsdl: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.

### 3.2.6.3 Create a Communication Arrangement for Inbound Communication with Service Key Basic

Learn how to quickly create a communication user and communication arrangement for an inbound communication scenario by using the basic service key.

#### Prerequisites

- You have created a communication scenario.
- You have created a space. See [Create Spaces \[page 922\]](#).
- 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 to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#).
2. In the navigation area, select **Spaces** and choose your space.
3. Select **Service Instances** from the navigation area and choose your ABAP service instance.
4. In the navigation area, choose **Service Key** and click on the **Create Service Key** button.
5. In the **Create Service Key** dialog, enter a name for your communication scenario and fill in the **Configuration Parameters (JSON)** field as follows:

```
{  
    "scenario_id": "SAP_COM_XYZ",  
    "type": "basic"  
}
```

### i 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.

## 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.

### 3.2.6.4 SOAP Communication

SOAP-based web service outbound communication within the ABAP environment is enabled by using SOAP destinations in a corresponding destination service.

## Context

The ABAP environment allows you to use synchronous outbound web services based on the SOAP protocol. These SOAP services can be used for the communication with S/4HANA Cloud, on-premise systems, or any other SOAP service exposed to the Internet.

The configuration of SOAP web services in the ABAP environment can be done with the help of SOAP destinations.

A SOAP destination is used to instantiate the web service consumer proxy in the coding.

You have the following options to create SOAP destinations:

- **URL approach:** By using a plain URL and setter methods to provide the needed configuration directly in the coding.
- **Destination service approach:** By using a SOAP destination, which is maintained in a destination service that resides in the Cloud Foundry environment. See [Consuming the Destination Service..](#)

### i Note

In on-premise systems, SOAP services are configured by logical ports, which are maintained in transaction SOAMANAGER. In the ABAP environment, this concept of logical ports was replaced by the concept of SOAP destinations.

## Use

### Cloud Destination

#### ↳ Sample Code

```
TRY.  
    DATA(lo_soap_dest) =  
        cl_soap_destination_provider->CREATE_BY_CLOUD_DESTINATION(  
            i_name           = 'test_destination'  
            i_service_instance_name = 'DESTINATIONSERVICE'  
        ).  
    DATA(lo_consumer) = new CO_FOO_PROXY( destination = lo_dest ).  
    lo_consumer->foo_method(  
        EXPORTING  
            input = l_request  
        IMPORTING  
            output = DATA(l_response)  
    ).  
    CATCH cx_soap_destination_error INTO DATA(lx_error).  
        error_handling( lx_error ).  
    ENDTRY.
```

### Plain URL

#### ↳ Sample Code

```
TRY.  
    DATA(lo_dest) = cl_soap_destination_provider->CREATE_BY_URL(  
        i_url = 'https://foo_host/sap/bc/srt/rfc/sap/foo_provider/000/  
        foo_service/foo_binding' ).  
    lo_dest->set_basic_authentication( i_user = p_user i_password =  
        p_pwd ).  
    DATA(lo_consumer) = new CO_FOO_PROXY( destination = lo_dest ).  
    lo_consumer->foo_method(  
        EXPORTING  
            input = l_request  
        IMPORTING  
            output = DATA(l_response)  
    ).  
    CATCH cx_soap_destination_error cx_ai_system_fault INTO DATA(lx_error).  
        error_handling( lx_error ).  
    ENDTRY.
```

You can set further web service properties in the service destination under *Additional properties*:

- `ws.soapVersion`: SOAP version
- `ws.maxWaitTime`: Maximum wait time for the consumer (in seconds)
- `ws.compressMessage`: Enables compression of message
- `ws.soapAction.<operationName>`: Sets the SOAP action for a given operation with name `<operationName>`

Instead of setting these properties in the service destination, you can set them programmatically by using the setter method provided in interface `IF_SOAP_DESTINATION`.

- `set_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( )`: Maximum wait time for the consumer (in seconds)
- `set_compress_message( )`: Enables compression of message
- `set_soap_action( )`: SOAP action for a given operation, parameters are the name of the operation and the value of the SOAP action
- `set_url( )`: URL of the endpoint
- `set_basic_authentication( )`: Sets user and password

#### i Note

Currently, only basic authentication is supported. You can set a user and password for the destination with method `if_soap_destination=>set_basic_authentication( )`.

## Related Information

[Consuming a Web Service \[page 439\]](#)

### 3.2.6.4.1 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. This is the prerequisite for the ILM process that physically deletes business users.

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
PersonExternalID	Person External ID Mandatory for business partner category role BUP003 (Employee) at creation.	60	0..1
PersonID	Person ID At least one of the person IDs is mandatory.	10	0..1
PersonUUID	Person UUID At least one of the person IDs is mandatory.	36	0..1

Node or Field	Description	Maximum Field Length	Cardinality
BusinessPartnerRoleCode	Business Partner Role Code	6	0..1
	Only business partner role code BUP003 (Employee) is supported.  This field is mandatory.		
MarkedForArchivingIndicator	Mark for Archiving	0..1	
	Set to <b>True</b> :		
	<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>		
	Set to <b>False</b> :		
	<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>		
ValidityPeriod StartDate	Format:	0..1	
Cardinality: 0..1	YYYY-MM-DD		
	By default, the system date is set.		
EndDate	Format:	0..1	
	YYYY-MM-DD		
	By default, 9999-12-31 is set.		

<b>Node or Field</b>		<b>Description</b>	<b>Length</b>	<b>Maximum Field Cardinality</b>
PersonalInformation	FormOfAddress	Form of address	4	0..1
Cardinality: 0..1	FirstName	First name	40	0..1
	LastName	Last name	40	0..1
		This field is mandatory.		
	PersonFullName	Person full name	80	0..1
	AcademicTitle	Academic title	4	0..1
	CorrespondenceLanguage	Correspondence language	9	0..1
	MiddleName	Middle name	40	0..1
	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
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

Node or Field		Description	Maximum Field Length	Cardinality
User( <b>only for Cloud</b> )	UserName	User name/Alias	40	0..1
	LogonLanguageCode	Logon language	9	0..1
Cardinality: 0..1				

Node or Field	Description	Maximum Field Length	Cardinality
DateFormatCode	<p>You can use the following values:</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	Length	Maximum Field Cardinality
	<ul style="list-style-type: none"> <li>• C - YYYY/MM/D D (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 - 1 234 567,89</li> </ul>	2	0..1
TimeZoneCode	Time zone	10	0..1
TimeFormatCode	You can use the following values:  <ul style="list-style-type: none"> <li>• 0 - 24 Hour Format (Ex- ample: 12:05:10)</li> <li>• 1 - 12 Hour Format (Ex- ample: 12:05:10 PM)</li> <li>• 2 - 12 Hour Format (Ex- ample: 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 StartDate	Format: YYYY-MM-DD  If no start date is maintained for the User, the StartDate for the BusinessUser is entered.	1	
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	actionCode	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.	2 optional
	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 [3] is not set and user data (UserName and Role) are given.	2 optional
[6] roleListCompleteTransmissionIndicator	CTI for the Role node		optional

Node or Field		Description	Length	Maximum Field Cardinality
UserAssignment <b>(only for on-premises)</b> Cardinality: 0..1	UserID	User ID	12	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
WorkplaceInformation Cardinality: 0..1	EmailAddress	Email address	241	0..1
	PhoneInformation Cardinality: 0..2	<p>Phone type</p> <ul style="list-style-type: none"> <li>• B - Business</li> <li>• C - Cell</li> </ul> <p>One set of phone information per phone type supported.</p>	1	1
	CountryDialingCode	<p>Country dialing code</p> <p>Used for both phone types.</p>	10	0..1
	PhoneNumberAreaID	<p>Phone number area code</p> <p>Used for phone type B only.</p>	10	0..1
	PhoneNumberSubscriberID	<p>Phone number subscriber ID</p> <p>Used for both phone types.</p>	30	0..1
	PhoneNumberExtension	<p>Phone number extension</p> <p>Used for phone type B only.</p>	10	0..1

Node or Field		Description	Maximum Field Length	Cardinality
	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 [7] is not set and phone data is given.</p>	2	optional
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
	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
[7] phoneInformationListCompleteTransmissionIndicator		CTI for the PhoneInformation node		optional
[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>	2	optional
[2] personalInformationListCompleteTransmissionIndicator		CTI for the PersonalInformation node		optional

Node or Field	Description	Length	Maximum Field Cardinality
[3] userListCompleteTransmissionIndicator	CTI for the User node		optional
[4] userAssignmentListCompleteTransmissionIndicator	CTI for the UserAssignment node		optional
[5] workplaceInformationListCompleteTransmissionIndicator	CTI for the WorkplaceInformation node		optional

## 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>
          <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>
        </WorkplaceInformation>
      </BusinessUser>
    </aba:BusinessUserBundleMaintainRequest_sync>
  </soapenv:Body>
</soapenv:Envelope>

```

```

<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</PhoneNumberAreaID>
                        <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		Description	Maximum Field 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	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	0..1
Cardinality: 0..unbounded	CategoryCode	Not in use	15	0..1
	SeverityCode	Severity code definition:	1	0..1
		<ul style="list-style-type: none"> <li>• 1 - Information</li> <li>• 2 - Warning</li> <li>• 3 - Error</li> </ul>		

Field or Node	Description	Maximum Field Length	Cardinality
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><code>PersonExternalID</code> and <code>PersonID</code> have a 1:1 relationship.</p> <p>Enter the <code>PersonID</code> that corresponds with the <code>PersonExternalID</code>.</p>
105	<p>Combination of Ext. ID &amp;1 and UUID &amp;2 inconsistent.</p> <p><code>Person ExternalID</code> and <code>PersonUUID</code> have a 1:1 relationship.</p> <p>Enter the <code>PersonUUID</code> that corresponds with the <code>PersonExternalID</code></p>

## Constraints

This service does not support:

- Service Performer (BBP005) business users
- Freelancer (BBP010) business users

## Additional Information

### i Note

For more information about the API, choose the [Details](#) tab on the SAP API Business Hub.

For more details about Communication Management, see [Communication Management \[page 1017\]](#).

## 3.2.6.4.2 Business User - Read

Technical name: QUERYBUSINESSUSERIN

This synchronous inbound SOAP service enables you to provision 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
PersonExternalIdInterval	<p>IntervalBoundaryTypeCode</p> <p>You can use the following values:</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

Field or Node	Description	Maximum Field Length	Cardinality
	UpperBoundaryPers onExtId	60	0..1
PersonIDInterval Cardinality: 0..un- bounded	IntervalBoundaryT ypeCode  You can use the following values: <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> This field is mandatory if LowerBoundaryPers onId is set.	1	1
LowerBoundaryPers onId		10	0..1
UpperBoundaryPers onId		10	0..1

Field or Node	Description	Maximum Field Length	Cardinality
BusinessPartnerRoleCodeInterval	<p>IntervalBoundaryTypeCode</p> <p>You can use the following values:</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
MarketForArchivingIndicator	<p>IntervalBoundaryTypeCode</p> <p>You can use the following values:</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..unbounded	<p>IntervalBoundaryTypeCode</p> <p>You can use the following values:</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 LowerBoundaryUserId is set.</p>	1	1
LowerBoundaryUserId		12	0..1
UpperBoundaryUserId		12	0..1

Field or Node	Description	Maximum Field Length	Cardinality
UserNameInterval Cardinality: 0..un- bounded	<p>IntervalBoundaryT ypeCode</p> <p>You can use the following values:</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>You can use the following values:</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..unbounded	<p>IntervalBoundaryTypeCode</p> <p>You can use the following values:</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 LowerBoundaryLast Name is set.</p>	1	1
LowerBoundaryLastName		40	0..1
UpperBoundaryLastName		40	0..1
EmailAddressInterval Cardinality: 0..unbounded	<p>IntervalBoundaryTypeCode</p> <p>LowerBoundaryEmailAddress</p> <p>UpperBoundaryEmailAddress</p>	1 241 241	1 0..1 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	You can use the following values: <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	You can use the following values: <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	You can use the following values: <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/">
  <ns1:aba="http://sap.com/xi/ABA">
    <soapenv:Header/>
    <soapenv:Body>
      <ns1:BusinessUserSimpleByElementsQuery_sync>
        <ns1:BusinessUser>
          <ns1:PersonIDInterval>
            <ns1:IntervalBoundaryTypeCode>1</ns1:IntervalBoundaryTypeCode>
            <!--Optional:-->
            <ns1:LowerBoundaryPersonID>9980035943</ns1:LowerBoundaryPersonID>
            <!--Optional:-->
          </ns1:PersonIDInterval>
          <ns1:BusinessPartnerRoleCodeInterval>
            <ns1:IntervalBoundaryTypeCode>1</ns1:IntervalBoundaryTypeCode>
            <!--Optional:-->
            <ns1:LowerBoundaryBusinessPartnerRoleCode>bup003</ns1:LowerBoundaryBusinessPartnerRoleCode>
            <!--Optional:-->
          </ns1:BusinessPartnerRoleCodeInterval>
        </ns1:BusinessUser>
      <ns1:QueryProcessingConditions>
        <!--Optional:-->
      </ns1:QueryProcessingConditions>
    </soapenv:Body>
  </ns1:aba>
</soapenv:Envelope>

```

```

<QueryHitsMaximumNumberValue>1</QueryHitsMaximumNumberValue>
<QueryHitsUnlimitedIndicator>false</QueryHitsUnlimitedIndicator>
</QueryProcessingConditions>
</aba:BusinessUserSimpleByElementsQuery_sync>
</soapenv:Body>
</soapenv:Envelope>

```

## Service Response

### Business User (BusinessUser)

**i 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	StartDate Cardinality: 0..1 Format: YYYY-MM-DD	1	
	EndDate Format: YYYY-MM-DD	1	
PersonalInformation	FormOfAddress Cardinality: 0..1 FirstName LastName PersonFullName AcademicTitle CorrespondenceLanguage	4 40 40 80 4 9	0..1 0..1 0..1 0..1 0..1 0..1

Node or Field		Description	Length	Maximum Field Cardinality
	MiddleName	Middle name	40	0..1
	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
User	UserID	User ID	12	1
Cardinality: 0..1	UserName	User name/Alias	40	1
	LogonLanguageCode	Logon language	9	0..1

Node or Field	Description	Maximum Field Length	Cardinality
DateFormatCode	<p>You can use the following values:</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	Length	Maximum Field Cardinality
	<ul style="list-style-type: none"> <li>• C - YYYY/MM/D D (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 - 1 234 567,89</li> </ul>	2	0..1
TimeZoneCode	Time zone	10	0..1
TimeFormatCode	You can use the following values:  <ul style="list-style-type: none"> <li>• 0 - 24 Hour Format (Ex- ample: 12:05:10)</li> <li>• 1 - 12 Hour Format (Ex- ample: 12:05:10 PM)</li> <li>• 2 - 12 Hour Format (Ex- ample: 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	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	Role name	40	1	
RoleName	Role name	40	1	
Cardinality: 0..unbounded				
UserAssignment	User ID	12	1	
UserAssignment	UserName	40	0.1	
WorkplaceInformation	EmailAddress	Email address	241	0.1
WorkplaceInformation	PhoneInformation	Phone type	1	1
WorkplaceInformation	CountryDialingCode	Country dialing code	10	0.1
WorkplaceInformation	PhoneNumberAreaID	Phone number area code	10	0.1
WorkplaceInformation	PhoneNumberSubscriberID	Phone number subscriber ID	30	0.1
WorkplaceInformation	PhoneNumberExtension	Phone number extension	10	0.1
FunctionalTitleName	Functional title name	40	0.1	
Department	Department name	40	0.1	
RoomNumber	Room number	10	0.1	

Node or Field	Description	Maximum Field Length	Cardinality
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 found 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 maximum of all dropped severity codes worst level will be shown.	1	0..1	
Item	TypeID	Message number	40	0..1
Cardinality: 0..unbounded	CategoryCode	Not in use	15	0..1

Field or Node	Description	Maximum Field Length	Cardinality
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:

- Service Performer (BBP005) business users
- Freelancer (BBP010) business users

## Additional Information

### i Note

For more information about the API, choose the [Details](#) tab on the SAP API Business Hub.

For more details about Communication Management, see [Communication Management \[page 1017\]](#).

## 3.2.6.5 HTTP Communication

Use the HTTP client to enable HTTP communication from the SAP Cloud Platform ABAP environment.

## Context

The HTTP client, which is whitelisted for the SAP Cloud Platform ABAP environment, is a wrapper of the well-known (but not whitelisted) ABAP HTTP client.

It is used for communication with S/4HANA Cloud, on-premise systems, or any other HTTP service exposed to the Internet.

The HTTP client provides integration with, for example, the Destination service residing in the SAP Cloud Platform Cloud Foundry environment.

See also [Consuming the Destination Service](#).

As there are no SM59-managed destinations (destinations created via SAP transaction SM59) available in the ABAP environment, the decoupling of designtime and runtime is done via generic HTTP destinations that you can create using a plain URL or a Cloud Foundry destination:

#### ↳ Sample Code

```
DATA(lo_url_destination) =  
cl_http_destination_provider=>create_by_url( 'https://foo.bar' ).  
  
DATA(lo_cloud_destination) =  
cl_http_destination_provider=>create_by_cloud_destination(  
i_name = 'S4Demo'  
).
```

#### i Note

Typically, you will be using the destinations of the subaccount in which the ABAP instance resides. 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.

## Use

The actual **processing of an HTTP request** and its response is shown in the following code example, where an S/4HANA OData API is called:

#### ↳ 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 URL  
    lo_http_destination =  
    cl_http_destination_provider=>create_by_url( 'https://my300098-  
api.s4hana.ondemand.com/sap/opu/odata/sap/API_BUSINESS_PARTNER' ). "/  
A_BusinessPartner' ).  
  
    " 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 )  
  
    ( name = 'APIKey' value = '<API_KEY>' ) ) ).
```

```

    " Adding Authorization Information
    lo_http_client->get_http_request( )-
>set_authorization_basic( i_username = 'INBOUND_COM_USER' i_password =
'someseccurepassword' ).

    " 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.

```

## i Note

Instead of providing a hard-coded password, you can also use the Destination service to retrieve the required information (recommended).

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.
  WRITE 'An exception has occurred!'.
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.

```

## Related Information

[Connectivity in the Cloud Foundry Environment](#)

## 3.2.6.6 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](#)

## 3.2.6.7 RFC Communication

Use the `CALL FUNCTION ... DESTINATION` statement with a destination of proxy type `Internet` that is defined in the Destination service to call other systems via WebSocket RFC from the ABAP environment.

### Context

Using outbound calls via WebSocket RFC from the ABAP environment, you can reach remote-enabled function modules in SAP cloud products and on-premise systems that are exposed to the internet. Currently, basic authentication is supported as authentication type.

### Prerequisites

- The respective destination exists in the relevant instance of the Destination service. The destination must be of type `RFC` and use the proxy type `Internet`.

#### i Note

Since there are no SM59-managed destinations (destinations created via SAP transaction SM59) available in the ABAP environment, the decoupling of design time and runtime is done through the Destination service.

For more information, see [Integrating On-Premise Systems \[page 478\]](#).

### Example

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
DATA(lr_dest) = cl_rfc_destination_provider->create_by_cloud_destination(
    EXPORTING
        i_name           = 'name_of_destination'
).
DATA(lv_dest) = lr_dest->get_destination_name( ).
"Using the destination in a Remote Function Call
CALL FUNCTION 'RFCPING' DESTINATION lv_dest.
```

#### i Note

Typically, you will be using the destinations of the subaccount in which the ABAP instance resides. 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.

### 3.2.6.8 Develop a Remote-Enabled Function Module (RFM)

Develop RFMs in ABAP Development Tools (ADT) from the ABAP environment.

In the ABAP environment, you can develop RFMs in *ABAP Development Tools* (ADT) in Eclipse. For more information, see [ABAP Development User Guide](#).

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 in SAP Cloud Platform are created automatically, such as authorization default values and the respective inbound service.

For detailed information on consuming an RFM in an inbound communication scenario, see [Set Up an Inbound RFC Connection \[page 485\]](#).

### 3.2.6.9 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.

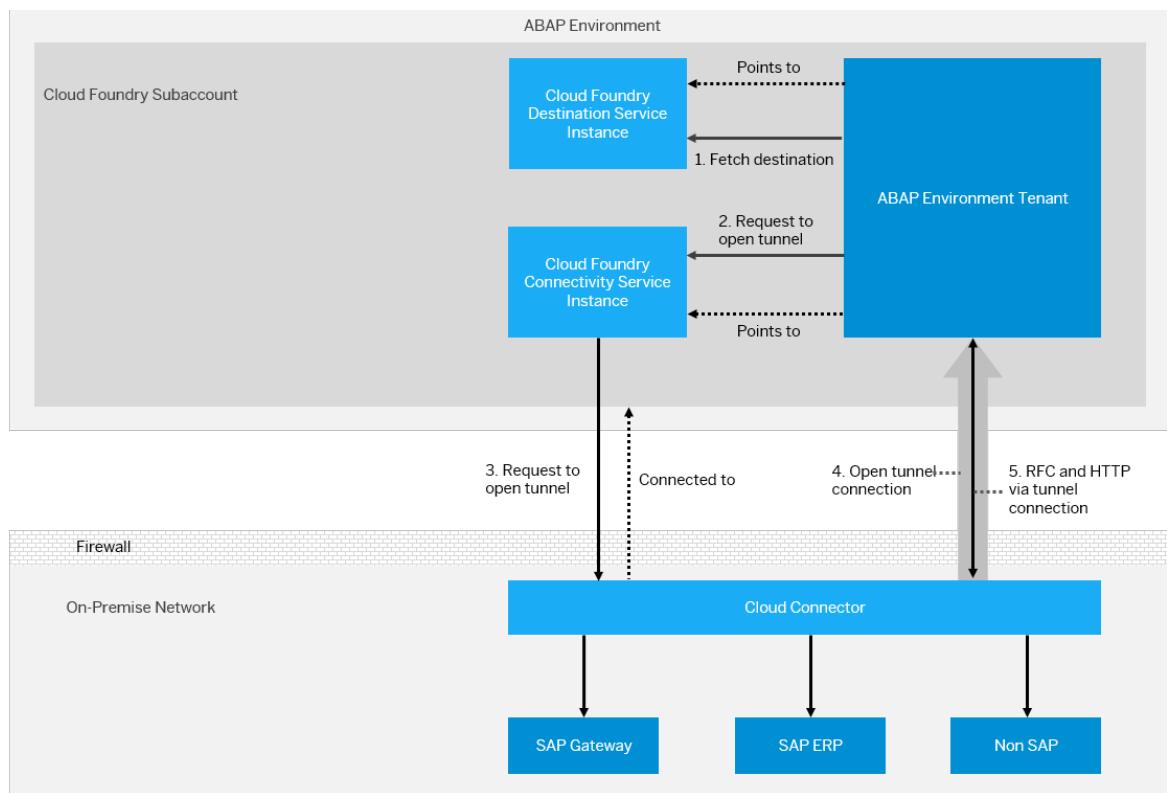
For more information on the deprecated scenario that uses the Connectivity service in the Neo environment, see [Create a Communication Arrangement for Cloud Connector Integration \(Deprecated\) \[page 480\]](#).

#### i Note

In this default scenario, you must connect the Cloud Connector to and configure trust for your *Cloud Foundry* subaccount as described in [SAP Cloud Platform Connectivity: Cloud Connector](#). 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**.

After you have completed this setup, a connection from the ABAP environment tenant to an on-premise system is established in the following order (the picture below shows the default scenario):

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 RFC and HTTP destinations. See [Setting Up Destinations to Enable On-Premise Connectivity \[page 482\]](#).

### 3.2.6.9.1 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

##### i Note

This procedure is deprecated. 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 SAP Cloud Platform Neo subaccount name.
- The name of the region host of your Neo subaccount

##### i 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](#)

##### i 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.

### i 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 URL. If your Neo URL is <a href="https://account.xyz.hana.ondemand.com">https://account.xyz.hana.ondemand.com</a> , enter account.
<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</i> , select <i>Cloud Connector</i> . d) Enter the user name and password of the user you created.

For more information on how to create a communication system, see [How to Create Communication Systems \[page 1024\]](#).

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.
► <i>Additional Properties</i> ► <i>Account Name</i> ▶	Enter the name of your Neo subaccount.
► <i>Outbound Communication</i> ► <i>User Name</i> ▶	

For more information on how to create a communication arrangement, see [How to Create a Communication Arrangement \[page 1022\]](#).

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 [Setting Up Destinations to Enable On-Premise Connectivity \[page 482\]](#).

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.

### 3.2.6.9.2 Setting Up Destinations to 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:

- [Set Up an HTTP Destination \[page 483\]](#)
- [Set Up an RFC Destination \[page 484\]](#)

### 3.2.6.9.2.1 Set Up an HTTP Destination

Set up on-premise HTTP connectivity for the SAP Cloud Platform ABAP environment by configuring an HTTP destination of proxy type *OnPremise*.

To configure an HTTP destination in the SAP Cloud Platform cockpit, perform the following steps:

1. Navigate to the relevant destination service instance.

#### i Note

Instead of creating a destination from your own Destination service instance, you can configure destinations directly on subaccount level, in the subaccount in which the ABAP instance resides (recommended). See also [Integrating On-Premise Systems \[page 478\]](#).

2. In the menu, navigate to *Destinations*.
3. Select *New Destination*.
4. In the *Destination Configuration* section, use the value help to select *HTTP* as *Type*.
5. (Optional) If you are using more than one Cloud Connector in your subaccount, you must enter the *Location ID* of the target Cloud Connector.  
See also [Managing Subaccounts](#) (section **Procedure**, step 4).
6. For *Proxy Type*, select *OnPremise* from the value help.
7. For *Authentication*, select *BasicAuthentication* or *PrincipalPropagation*.
8. Fill in the required fields and select *Save*.
9. Open Eclipse, and create and execute a runnable class.
10. To enable the HTTP communication, use, for example, the following API.

#### Sample Code

```
DATA(lo_destination) =  
cl_http_destination_provider=>create_by_cloud_destination(  
  i_name = 'ERP_HTTP'  
  i_authn_mode = if_a4c_cp_service=>service_specific  
 ).  
DATA(lo_client) =  
cl_web_http_client_manager=>create_by_http_destination( lo_destination ).  
DATA(lo_request) = lo_client->get_http_request( ).  
DATA(lo_response) = lo_http_client->execute( i_method =  
  if_web_http_client=>get ).  
out->write( lo_response->get_text( ) ).
```

#### i Note

*i\_name* is the name of the destination that you have configured in the previous steps.

#### i Note

Make sure that the called remote function module is exposed in Cloud Connector. See [Configure Access Control \(HTTP\)](#).

### 3.2.6.9.2.2 Set Up an RFC Destination

Set up on-premise RFC connectivity for the SAP Cloud Platform ABAP environment by configuring a destination of type RFC.

To configure an RFC destination in the SAP Cloud Platform cockpit, perform the following steps:

1. Navigate to the relevant destination service instance.

#### i Note

Instead of creating a destination from your own Destination service instance, you can configure destinations directly on subaccount level, in the subaccount in which the ABAP instance resides (recommended). See also [Integrating On-Premise Systems \[page 478\]](#).

2. In the menu, navigate to *Destinations*.
3. Create a destination by selecting *New Destination*.
4. For *Type*, select *RFC* from the value help.
5. (Optional) If you are using more than one Cloud Connector in your subaccount, you must enter the *Location ID* of the target Cloud Connector.  
See also [Managing Subaccounts](#) (section **Procedure**, step 4).
6. As authentication type, use basic authentication or set property  
`jco.destination.auth_type=PrincipalPropagation`.
7. If you use basic authentication, set a user name and credentials for the destination.
8. To configure the RFC destination, choose either of the following options:
  - For a destination that uses **load balancing** (system ID and message server), proceed as follows:
    1. Select *New Property*, choose `jco.client.r3name` from the value help and enter the three-letter system ID of your backend system (as configured in Cloud Connector) in the property field.
    2. Create another property, select `jco.client.mshost`, and enter the message server host (as configured in Cloud Connector) in the property field.
    3. Add another property, choose `jco.client.group`, and enter a log group in the property field.
    4. Create another property, select `jco.client.client`, and enter the three-digit ABAP client number.
  - For a destination **without load balancing** (application server and instance number), perform these steps:
    1. Select *New Property*, choose `jco.client.ashost` from the value help and enter the application server name of your backend system (as configured in the Cloud Connector) in the property field.
    2. Add another property, choose `jco.client.sysnr`, and enter 00, the instance number of the application server (as configured in Cloud Connector) in the property field.

3. Create another property, select `jco.client.client`, and enter the three-digit ABAP client number.
9. Select [Save](#).

## Next Step (Optional): Call a Remote Function Module

1. Open Eclipse, and create and execute a runnable class.
2. To execute a remote function module on your on-premise system, use the following code snippet:

### ↳ Sample Code

```
DATA(lo_destination) =  
cl_rfc_destination_provider->create_by_cloud_destination(  
  i_name = 'ERP_RFC'  
).  
DATA(lv_destination) = lo_destination->get_destination_name( ).  
DATA lv_result type c length 200.  
CALL function 'RFC_SYSTEM_INFO'  
DESTINATION lv_destination  
  IMPORTING  
    rfcси_export = lv_result.  
  out->write( lv_result ).
```

### ℹ Note

`i_name` is the name of the destination that you have configured in the previous steps.

### ℹ Note

Make sure that the called remote function module is exposed in Cloud Connector. See [Configure Access Control \(RFC\)](#).

### 3.2.6.9.3 Set Up an Inbound RFC Connection

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.

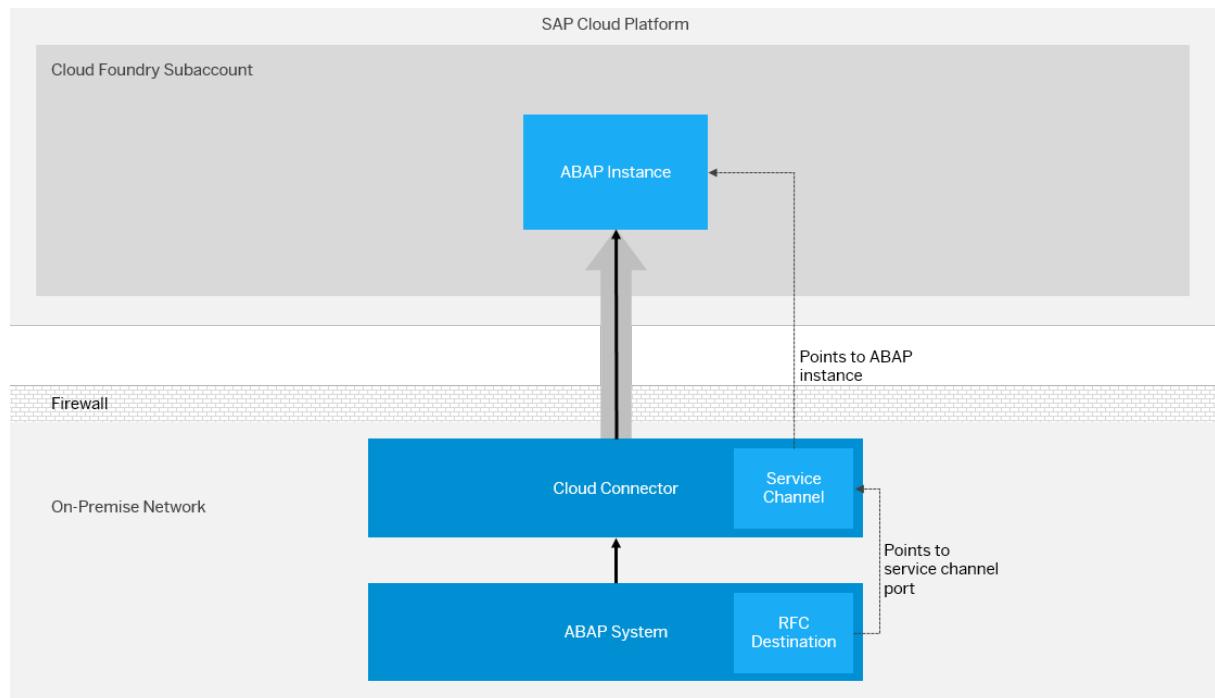
### Context

RFC connections from on-premise systems to the ABAP environment can be performed through a so called Cloud Connector service *channel*.

### i Note

As of SAP S/4HANA version 2019, you can 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 the ABAP environment on SAP Cloud Platform.



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 instance number.

### i Note

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.

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](#).

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.
  - Use the connection test in transaction SM59 to verify that the service channel works correctly.

## 3.2.7 Released Components and Objects

### 3.2.7.1 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**.

#### 3.2.7.1.1 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).

### 3.2.7.1.2 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](#).

### 3.2.7.1.3 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](#).

### 3.2.7.1.4 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).

### 3.2.7.1.5 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](#).

### **3.2.7.1.6 Date/Time and Time Stamp Processing**

To handle and convert time stamps, use the following classes:

- `CL_ABAP_DTFM`
- `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.

### **3.2.7.1.7 Parallel Processing**

To start parallel tasks, use class `CL_ABAP_PARALLEL`.

For more information, see the ABAP Doc comments of this class in ABAP Development Tools (ADT).

### **3.2.7.1.8 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).

### **3.2.7.1.9 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).

### 3.2.7.1.10 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).

### 3.2.7.1.11 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).

### 3.2.7.1.12 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](#).

### 3.2.7.1.13 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`.

#### i 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.

### 3.2.7.2 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* (which is an object class). 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.

#### i 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.

### 3.2.7.2.1 Authorization Checks

#### Authorization Check for Change Document Object Maintenance

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.

## • Example

### ↳ 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'
        it_activity    = '03'
        it_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 `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.

## 3.2.7.2.2 Managing Change Document Objects

Use method `CL_CHDO_OBJECT_TOOLS_REL` to maintain change document objects.

For more information, see

- [Creating a Change Document Object \[page 493\]](#)
- [Updating a Change Document Object \[page 498\]](#)
- [Deleting a Change Document Object \[page 502\]](#)
- [Reading a Change Document Object Definition \[page 504\]](#)

### 3.2.7.2.2.1 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.
- Enter a name space separately from the name of the object without name space. Once you have done so, the name space and the object name will always be displayed together. All generated objects for this change document object will automatically be generated within the name space.
- 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, a class (name granted automatically `CL_<change document object name>_CHDO`) with methods `WRITE` and `IF_CHDO_ENHANCEMENTS~AUTHORITY_CHECK` is created. `IV_CL_OVERWRITE` can be used to specify whether an existing class with the specified name can be overwritten or not. Changes are saved in the transport request (`IV_CORRNR`).

The export parameter `ET_ERRORS` is used to return all generation messages (in the 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

Parameter Name	Field Name	Value Help
	TABNAME	Name of the table as defined in the dictionary
	MULTCASE	Set this flag if the change data is to be passed in an internal table (multiple case). If you do not select this field, the change data will be passed in a work area (single case).
	DOCUDEL	<p>DOCUDEL = SPACE</p> <p>The deletion of a table entry will be documented in one change document position 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 position.</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 position 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 position.</p> <p>The change indicator is 'J' instead of 'I'.</p>

Parameter Name	Field Name	Value Help
	REFNAME	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.  In single case, the reference information is passed as two additional work areas (old, new). In multiple case, the import tables (old, new) are enhanced to include the referenced structure.
	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

Parameter Name	Field Name	Value Help
	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.
	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 of generated class
ET_SYNT_ERROR_LONG		Syntax errors of generated class long text

## ↳ 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.
```

```
ENDIF.  
ENDMETHOD.  
ENDCLASS.
```

### 3.2.7.2.2 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 `TCOB` and `TADIR` and the generation information is read directly from database table `TCDRP`. If no generation information exists in table `TCDRP`, it can also be passed using import parameter `IS_CD_OBJECT_GEN`. In this case, 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 granted automatically `CL_<change document object name>_CHDO`) with methods `WRITE` and `IF_CHDO_ENHANCEMENTS~AUTHORITY_CHECK` is created. You can use `IV_CL_OVERWRITE` to specify whether an existing class can be overwritten with the specified name or not. Changes made to the class are saved in the transport request (`IV_CORRNR`).

The export parameter `ET_ERRORS` is used to return all generation messages (in the 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>	Set this flag if the change data is to be passed in an internal table (multiple case). If you do not select this field, the change data will be 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 position 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 position.</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 position 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 position.</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>In single case, the reference information is passed as two additional work areas (old, new). In multiple case, the import tables (old, new) are enhanced to include the referenced structure.</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 of generated class
ET_SYNT_ERROR_LONG		Syntax errors of generated class long text

### ↳ 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_tcdobt-tabname = 'ZCHDO'.
ls_tcdobt-multcase = 'X'.
ls_tcdobt-docudel = ''.
ls_tcdobt-docuins = ''.
ls_tcdobt-docud_if = ''.
APPEND ls_tcdobt TO p_it_tcdob.
ls_tcdobt-lang_key = 'D'.
ls_tcdobt-object_text = 'Single Case: Update'.
APPEND ls_tcdobt 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_crrorr          = '<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.

```

### 3.2.7.2.2.3 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`. If `SY_SUBRC = 0`. When this information is returned, the object was deleted.

Furthermore, the import parameter `IV_DEL_CL_WHEN_USED` determines if the class of the change document object is deleted (value `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 (in the message class CD).

#### Import Parameters

Parameter Name	Field Name	Value Help
IV_OBJECT		Change document object name
IV_CORRNR		Transport request where deletion should be logged
IV_DEL_CL_WHEN_USED		Delete generated class "CL_<change document object name>_CHDO" even if it is currently used.

#### 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

#### ↳ 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_corrhnr     = '<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.

```

### 3.2.7.2.2.4 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
	ET_OBJECT_INFO	Generation information type technical name (DEFINITION, GENERATION, CLASS)
	ET_OBJECT_INFO	Generation information type text (Definition, Generate, Class details)

Parameter Name	Field Name	Value Help
	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

#### ↳ 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.

```

### 3.2.7.2.3 Writing Change Documents

Write 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.

Import Parameters

Parameter Name	Value Help
OBJECTID	Change document object value
UTIME	Time of change
UPDATE	Date of change

Parameter Name	Value Help
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	With this parameter you control whether the changes to be logged are real or planned changes.
NO_CHANGE_POINTERS	'X': no change pointers will be written
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> or Y<table name>	Workarea with original content
N_<table name> or X<table name>	Workarea with changed content
Export Parameter	
Parameter Name	Value Help
CHANGENUMBER	Change number of the document

### 3.2.7.2.4 Reading Change Documents

You can use class `CL_CHSO_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
I_OBJECTCLASS		
IT_OBJECTID		
I_DATE_OF_CHANGE		
I_TIME_OF_CHANGE		
I_DATE_UNTIL		
I_TIME_UNTIL		
IT_USERNAME		
IT_READ_OPTIONS		
	local_time	Use local time
	time_zone	Time zone used for reading
	it_changenr	Range table for change document number

#### Export Parameter

Parameter Name	Field Name	Value Help
ET_CDREDADD_TAB		Table type for structure CDREDADD, change documents return table

#### ↳ 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.

```

### 3.2.7.3 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 [ABAP CDS - Conversion Functions for Units and Currencies](#) 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 module BAPI\_EXCHRATE\_CREATEMULTIPLE.

API class CL\_EXCHANGE\_RATES provides a PUT method to write exchange rates to the corresponding customizing persistence. For more information, see the ABAP Doc comments in the class implementation.

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

```

### 3.2.7.4 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.

### i 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.

## 3.2.7.4.1 Maintaining Number Range Objects

Class `CL_NUMBERRANGE_OBJECTS` provides methods for maintaining number range objects.

For more information, see

- [Creating Number Range Objects \[page 509\]](#)
- [Changing Number Range Objects \[page 511\]](#)
- [Deleting Number Range Objects \[page 512\]](#)
- [Reading Number Range Objects \[page 513\]](#)

### 3.2.7.4.1.1 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).

Parameter Name	Field Name	Value Help
	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.
		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

Parameter Name	Field Name	Value Help
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_corrnr      TYPE cl_numberrange_objects=>nr_attributes-corrnr.
      lv_object      = 'Z TEST_03'.
      lv_devclass    = 'Z_SNUM'.
      lv_corrnr      = 'SIDK123456'.
    TRY.
      cl_numberrange_objects=>create(
        EXPORTING
          attributes = VALUE #( object      = lv_object
                                domlen     = 'CHAR8'
                                percentage = 5
                                buffer     = abap_true
                                noivbuffer = 10
                                devclass   = lv_devclass
                                corrnr     = lv_corrnr )
          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.

```

## 3.2.7.4.1.2 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_corrnr    = 'SIDK123456'.

...
cl numberrange_objects=>update(
  EXPORTING
    attributes = VALUE #( object      = lv_object
                           domlen     = 'CHAR8'
                           percentage = 9
                           buffer     = 'S'
                           noivbuffer = 12
                           devclass   = lv_devclass
                           corrnr     = lv_corrnr )
    obj_text   = VALUE #( object      = lv_object
                           langu      = 'E'
                           txt        = 'Update object'
                           txtshort   = 'Test update' )
  IMPORTING
    errors     = DATA(lt_errors)
    returncode = DATA(lv_returncode)
  ).

...
```

### 3.2.7.4.1.3 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
  ).

...
```

### 3.2.7.4.1.4 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.
	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.

Parameter Name	Field Name	Value Help
	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)
).
...
...
```

## 3.2.7.4.2 Maintaining Intervals of Number Range Objects

The class `CL_NUMBERRANGE_INTERVALS` provides methods for maintaining intervals of number range objects.

For more information, see

- [Creating Intervals of Number Range Objects \[page 514\]](#)
- [Changing Intervals of Number Range Objects \[page 516\]](#)
- [Deleting Intervals of Number Range Objects \[page 517\]](#)
- [Reading Intervals of Number Range Objects \[page 518\]](#)

### 3.2.7.4.2.1 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

Parameter Name	Field Name	Value Help
	FROMNUMBER	From Number
	TONUMBER	To Number
	NRLEVEL	Number Range Level
	EXTERNIND	Internal (' ') 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
	TABLENAMES	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 (' ') or external ('X') number range flag
	PROCIND	Processing flag (I=Insert, D=Delete, U=Update, '='no changes)

Parameter Name	Field Name	Value Help
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-nrrangenr  = '01'.
  ls_interval-fromnumber = '00000001'.
  ls_interval-tonumber   = '19999999'.
  ls_interval-procind    = 'I'.
  APPEND ls_interval TO lt_interval.
  ls_interval-nrrangenr  = '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.
...
```

## 3.2.7.4.2.2 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)
...
...
```

### 3.2.7.4.2.3 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-nrrangernr = '01'.
ls_interval-fromnumber = '00000001'.
ls_interval-tonumber = '19999999'.
ls_interval-procind = 'D'.
APPEND ls_interval TO lt_interval.
ls_interval-nrrangernr = '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).
...
...
```

### 3.2.7.4.2.4 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.
...
```

### 3.2.7.4.3 Getting Numbers from an Interval

The CL\_NUMBERRANGE\_INTERVALS 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
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

External 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).
...
...
```

### 3.2.7.5 Proxy API for SAP Cloud Platform Workflow

With this proxy API, you can start SAP Cloud Platform Workflows out of your ABAP Environment.

The overall starting point is the class CL\_SWF\_CPWD\_API\_FACTORY\_A4C. Once you have the API object of interface IF\_SWF\_CPWD\_API, see the ABAP documentation on supported actions.

#### Prerequisites

You've executed the integration steps in your ABAP environment. See [SAP Cloud Platform Workflow Integration \[page 522\]](#).

#### ↳ Sample Code

This coding sample shows how to start an SAP Cloud Platform Workflow.

```
TYPES: BEGIN OF ty_context,
        some_property TYPE string,
        END OF ty_context.
DATA(lo_cpwf_api) = cl_swf_cpwd_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 = 'myprocess'
    iv_context            = lv_context_json
    iv_retention_time     = 30
    iv_callback_class     = 'ZCL_SWF_CPWD_CALLBACK'
).
```

#### ↳ Sample Code

This coding sample shows a completion callback that includes reading the context.

```
METHOD if_swf_cpwd_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 Cloud 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 '<Your Workflow Definition ID>',
lc_event_def_id      TYPE
if_swf_cpwf_api=>cpwf_evt_def_id   VALUE '<Your 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.
```

## 3.2.7.5.1 SAP Cloud Platform Workflow Integration

You can enable the communication between the ABAP environment and SAP Cloud Platform Workflow.

This scenario enables the ABAP environment applications to extend their business processes by using the SAP Cloud Platform Workflow service in the Cloud Foundry environment. It allows the provisioning of an API in the ABAP environment to start and control workflow instances running on SAP Cloud Platform Workflow.

You set up the scenario using the following tasks:

1. [Create a Communication Arrangement Using a Service Key \[page 522\]](#) (recommended)
2. [Create an SAP Cloud Platform Destination \[page 523\]](#)
3. [Set Scopes for the Service Instance Using the Command Line Interface \[page 524\]](#)
4. [Enable the SAP Web IDE \[page 525\]](#)
5. [Maintain Business Roles and Business Users \[page 528\]](#)

### 3.2.7.5.1.1 Create a Communication Arrangement Using a Service Key

To connect the ABAP environment and SAP Cloud Platform, you need a communication arrangement.

#### Procedure

1. Get the service key.
  - a. In the SAP Cloud Platform 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 workflow service 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 Cloud 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 Cloud 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.

## 3.2.7.5.1.2 Create an SAP Cloud Platform Destination

On the SAP Cloud Platform, you need to create a destination to enable the communication to the ABAP environment.

### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your subaccount. See [Navigate to Orgs and Spaces](#).
2. Choose   **Destinations**.
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 <a href="#">HTTP</a> .
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 service 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 <a href="#">BasicAuthentication</a> .
User	Set the user to the client ID as specified for the inbound communication.
Password	Set the password to the secret.

4. Save your entries.

### 3.2.7.5.1.3 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

1. [Download and Install the Cloud Foundry Command Line Interface](#)
2. [Create a Service Instance of Workflow Service Using the Command Line Interface](#)
3. Add the new system environment variable `<CF_HOME>` that points to your CF path.
4. Create a file named `scopes.json`, and paste the following JSON file:

#### Sample Code

```
{
  "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"
  ]
}
```

5. Determine the API endpoint.
  - a. Navigate to your subaccount.
  - b. On the subaccount overview page under *Cloud Foundry*, there's the API endpoint URL.
6. Go to the command line.
  - a. Log in to your Cloud Foundry account:

↳ Sample Code

```
cf login
```

- b. Enter your credentials.
- c. Choose the correct subaccount.
- d. To update the service instance, enter the following command:

↳ Sample Code

```
cf update-service <SERVICE_INSTANCE_NAME> -c c:/temp/scopes.json
```

Use the name of the workflow service instance from your subaccount.

### 3.2.7.5.1.4 Enable the SAP Web IDE

The SAP Web IDE runs in the Neo environment. Execute the following steps to be able to deploy workflow definitions from the Neo environment to the subaccount in the Cloud Foundry environment.

#### Procedure

1. Create a subaccount in the Neo environment if you don't have one. See [Create a Subaccount in the Neo Environment](#).
2. In the navigation area, choose *Services*.
3. Search for *SAP Web IDE Full-Stack*.
4. Choose [Go to Service](#).
5. From the left sidebar, open the *Preferences* perspective by choosing (Preferences).
6. Choose *Cloud Foundry*.
7. Select `https://api.cf.sap.hana.ondemand.com` as the API endpoint.
8. Enter your credentials for SAP Cloud Platform.
9. Save your entries.
10. Choose *Extensions*, and search for the *Workflow Editor*.
11. Enable the *Workflow Editor* extension by using the toggle.
12. Choose [Save](#).
13. Reload SAP Web IDE by choosing [Refresh](#).

14. From the left pane, choose (Development) and navigate to the *Workspace* folder.
15. Choose *Project from Template*.

  - a. Select the *Cloud Foundry* environment.
  - b. Select *Multi-Target Application*, then choose *Next*.
  - c. Enter a project name, then choose *Next* and *Finish*.

16. Create a workflow.

  - a. Enter a module name.
  - b. Enter a Workflow name.
  - c. Choose *Finish*.

17. To define your workflow definition, see [Model Service Task to Complete the Workflow in the ABAP Environment \[page 527\]](#).
18. In the `mta.yaml` file a new service instance is mentioned, To deploy it to the existing service instance, replace the two occurrences a) `module>requires>name` and b) `resources>name`. Otherwise, you create a new service instance when you deploy.

```

mta.yaml *WFTest.workflow
1 ID: S4_TO_CPWF
2 _schema-version: '2.1'
3 version: 0.0.1
4 modules:
5   - name: WFModel
6     type: com.sap.application.content
7     path: WFModel
8   requires:
9     - name: <SERVICE_INSTANCE_NAME>
10    parameters:
11      content-target: true
12 resources:
13   name: <SERVICE_INSTANCE_NAME>
14   parameters:
15     service-plan: standard
16     service: workflow
17   type: org.cloudfoundry.managed-service
18

```

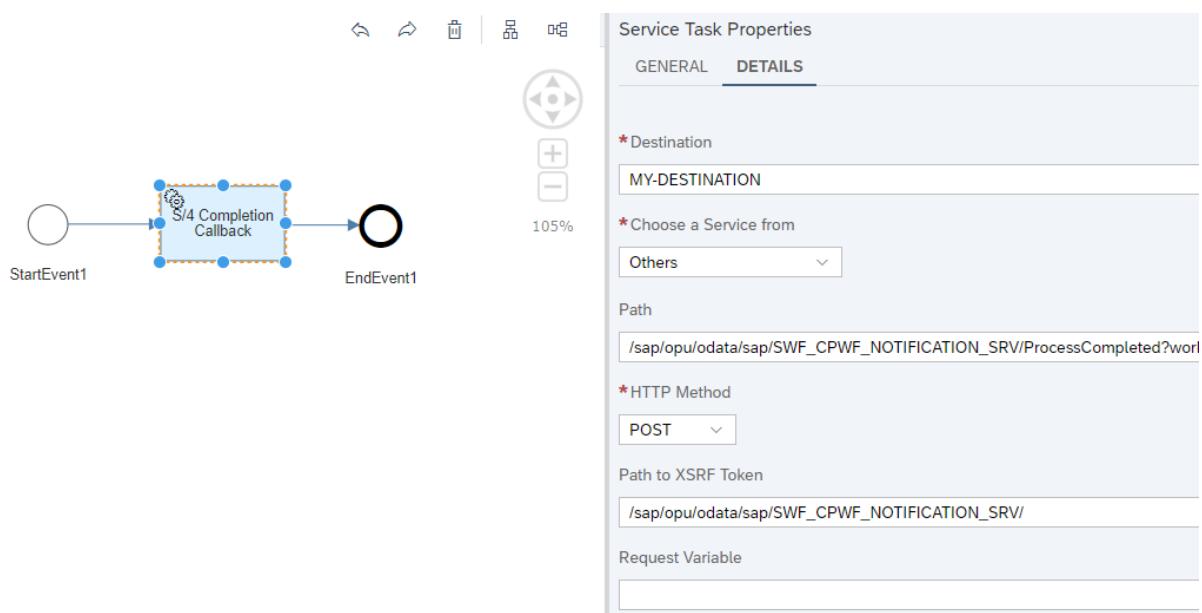
19. Build the project, right-click the project, and choose *Build*.
20. Deploy the project, open the `mta_archives` folder, right-click the `mtar` file, and choose *Deploy* .
21. If a popup appears and asks for the API endpoint and the service instance, select it again and choose *Deploy*.

### 3.2.7.5.1.5 Model Service Task to Complete the Workflow in the ABAP Environment

As the SAP Cloud Platform Workflow 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.

#### 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 SAP Cloud Platform Workflow API aren't communicated to the ABAP environment tenant.

#### Procedure

1. Open the workflow editor in SAP Web IDE.
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.

Field	Value
Choose a Service from	<i>Others</i>
Path	/sap/opu/odata/sap/SWF_CPFNOTIFICATION_SRV/ ProcessCompleted?workflowInstanceId='\\$ {info.workflowInstanceId}'
HTTP Method	<i>POST</i>
Path to XSRF Token	/sap/opu/odata/sap/SWF_CPFNOTIFICATION_SRV/

### 3.2.7.5.1.6 AIF Monitoring

AIF (Application Interface Framework) enables you to monitor inbound and outbound messages in the ABAP environment.

#### 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 \[page 528\]](#)

[Monitor Messages for SAP Cloud Platform Workflow Integration \[page 529\]](#)

### 3.2.7.5.1.6.1 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\_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.
3. With the [Maintain Business User](#) app, provide business users with access rights.

### **3.2.7.5.1.6.2 Monitor Messages for SAP Cloud Platform Workflow Integration**

You can monitor the background processes for SAP Cloud Platform Workflow integration.

#### **Prerequisites**

You have assigned business roles and business users.

#### **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 SAP Cloud Platform Workflow 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](#) 

### 3.2.7.6 Application Log

You can use application logs to display and check if any errors occurred during runtime.

For more information, see

- [Design Time API \[page 530\]](#)
- [Runtime API \[page 534\]](#)

#### 3.2.7.6.1 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.

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.
```

If the new application log object shall have subobjects, you can create these also 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 the 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.
```

## 3.2.7.6.1.1 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

### i 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
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

##### i 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.

## 3.2.7.6.2 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 \[page 535\]](#)
- [Set Header Information \[page 537\]](#)
- [Add Items \[page 538\]](#)
- [Writing Application Logs to the Database \[page 541\]](#)

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 \[page 544\]](#)
- [Read Application Logs from the Database \[page 545\]](#)
- [Get Header Information from the Log \[page 547\]](#)
- [Get Items from the Log \[page 548\]](#)

## 3.2.7.6.2.1 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 \[page 535\]](#)
- [Set Header Information \[page 537\]](#)
- [Add Items \[page 538\]](#)
- [Writing Application Logs to the Database \[page 541\]](#)

### 3.2.7.6.2.1.1 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 \[page 537\]](#).

#### • 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.

## ↳ 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.
...
```

### 3.2.7.6.2.1.2 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.
- 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.

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.

#### i 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.
...

```

### 3.2.7.6.2.1.3 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 \[page 539\]](#).
- Free Text (a text with a maximum length of 200 characters). For more information, see [Create a Free Text \[page 540\]](#).
- 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 \[page 541\]](#).

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 is 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**.

#### i Note

If an item was already added to the application log, its attributes cannot 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 probably be 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.
...
```

### 3.2.7.6.2.1.3.1 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'. 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.

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.

#### ↳ 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 ).
...

```

### 3.2.7.6.2.1.3.2 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 shall be visible.

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.

#### ↳ 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' ) .
...

```

### 3.2.7.6.2.1.3.3 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 shall be displayed.

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.

#### ↳ 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' ).
...
```

### 3.2.7.6.2.1.4 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 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.

To save an application log, the optional parameter `USE_2ND_DB_CONNECTION` is available. If the parameter is set, a second database connection is used for the saving. This second database connection is committed immediately after the save.

- `DELETE_LOG`: Deletes an application log from the database. The log is identified by the log object (which uses interface `IF_BAL_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_BAL_LOG`).
- `DEQUEUE`: Clear the enqueue which was set via method `ENQUEUE`. The log is identified by the log object (which uses interface `IF_BAL_LOG`).

### i 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( log = l_log,
                                              use_2nd_db_connection =
                                              abap_true ).

  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

### 3.2.7.6.2.2 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 544]
- Read Application Logs from the Database [page 545]
- Get Header Information from the Log [page 547]
- Get Items from the Log [page 548]

### 3.2.7.6.2.2.1 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.

#### i 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:

#### ↳ 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':

#### ↳ 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.
...
```

### 3.2.7.6.2.2.2 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 one application log from the database.
  - The log is identified by the log handle.
  - The method returns an object of interface `IF_BALI_LOG`.
  - If the optional parameter `READ_ONLY_HEADER` is set, only the log header is read from the database.  
The log items are not read yet.
- `LOAD_LOGS_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`.
  - If the optional parameter `READ_ONLY_HEADER` is set, 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: O3`

## • Example

Load a log from the database, generic code example:

### ↳ 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_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.
        ENDLOOP.
      ENDLOOP.
      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_via_filter( filter = l_filter ).
    CATCH cx_bali_runtime INTO DATA(l_exception).
        out->write( l_exception->get_text( ) ).
ENDTRY.
...
```

## 3.2.7.6.2.2.3 Get Header Information from the Log

An application log contains several header information and 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 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 it is not 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.

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.

### ❖ 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.
...
...
```

## 3.2.7.6.2.2.4 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.

Method of all log items is `GET_MESSAGE_TEXT`: It returns the message text of the item.

- 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

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 \[page 550\]](#).
- 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 \[page 550\]](#).
- 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 \[page 551\]](#).

## ❖ 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.
...

```

### 3.2.7.6.2.2.4.1 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.
...

```

### 3.2.7.6.2.2.4.2 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.
...

```

### 3.2.7.6.2.2.4.3 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.
...
...
```

### 3.2.7.6.2.3 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

## Exception Classes

If one of the class methods cannot be processed or cannot return the requested results, an exception is raised. The following exceptions are possible, each of them inherit from exception class CX\_BALI\_RUNTIME:

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.

Class Name	Description
CX_BALI_NOT_POSSIBLE	<p>The requested processing is not possible.</p> <p>Possible values of class attribute <code>ERROR_CODE</code>:</p> <ul style="list-style-type: none"> <li>• <code>CX_BALI_NOT_POSSIBLE=&gt;NO_AUTHORIZATION</code>: No authorization to access the log</li> <li>• <code>CX_BALI_NOT_POSSIBLE=&gt;OBJECT_NOT_ALLOWED</code>: Access to the log object is not allowed</li> <li>• <code>CX_BALI_NOT_POSSIBLE=&gt;TOO_MANY_ITEMS</code>: The maximum number 999999 of items was reached</li> <li>• <code>CX_BALI_NOT_POSSIBLE=&gt;SAVE_NOT_ALLOWED</code>: Error during saving to the database (e.g. database error or object is empty)</li> <li>• <code>CX_BALI_NOT_POSSIBLE=&gt;ENTRY_IS_LOCKED</code>: The enqueue cannot be set, because the log is already locked</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

#### i Note

Find more information about classes and interfaces of the [Application Log API](#) in the ABAB Development Tools (ADT).

### 3.2.7.6.2.3.1 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>	
<code>DB_HANDLER</code>	Database handler object: A reference to interface <code>IF_BALI_LOG_DB</code>

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>
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

### i 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>	

Name	Description
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

### i 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 does not 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.

Save a single log to the database:

SAVE\_LOG

Name	Description
<b>Importing parameters</b>	
LOG	Log which shall be saved: Reference to interface IF_BALI_LOG
USE_2ND_DB_CONNECTION	(Optional): If set, use 2nd database connection for saving Default: Not set
<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;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> </ul>

Name	Description
CX_BALI_INTERNAL_ERROR	Internal error during processing

**i Note**

- If a second database connection is used to save the log, a commit is executed on this 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> </ul>
CX_BALI_NOT_FOUND	The log was not found in the database
CX_BALI_INTERNAL_ERROR	Internal error during processing

**i 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>	

Name	Description
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 cannot be set, because the log is already locked</li> </ul>
CX_BALI_INTERNAL_ERROR	Internal error during processing

**i 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>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>OBJECT_NOT_ALLOWED: Access to the log object is not allowed
CX_BALI_INTERNAL_ERROR	Internal error during processing

**i Note**

If parameter LOG is initial, the method returns without exception.

### 3.2.7.6.2.3.2 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)
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	ERROR_CODE: CX_BALI_NOT_POSSIBLE=>OBJECT_NOT_ALLOWED: Access to the log object is not allowed

### i Note

- If parameter EXTERNAL\_ID is supplied and empty, the filter searches for logs with empty external identifier.
- If parameters OBJECT or SUBOBJECT 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)

Name	Description
USER_TABLE	(Optional): Table with log users (wildcards allowed)
<b>Returning parameter</b>	
NEW_FILTER	Reference to current filter object

**i Note**

If parameter USER is supplied and empty, it is 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
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)

Name	Description
USER_TABLE	
TIME_INTERVAL	Date and time interval
MAX_LOG_NUMBER	Maximum number of logs processed

### 3.2.7.6.2.3.3 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

Name	Description
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
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
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	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

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>	
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

### i 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 is 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 IF_BALI_ITEM_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 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

### i 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 BAPIRET-TAB
<b>Exceptions (inherit from CX_BALI_RUNTIME)</b>	
CX_BALI_NOT_POSSIBLE	ERROR_CODE: CX_BALI_NOT_POSSI-BLE=>TOO_MANY_ITEMS: The maximum number 999999 of items was reached
CX_BALI_INVALID_PARAMETER	The message ID of one of the messages is initial
CX_BALI_INTERNAL_ERROR	Internal error during processing

### i Note

If a message of the message table cannot be added to the log, it is 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 could not be added to the log.

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_INTERNAL_ERROR	Internal error during processing

Get all items from the log:

## GET\_ALL\_ITEMS

Name	Description
<b>Returning parameter</b>	
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_INTERNAL_ERROR	Internal error during processing

### 3.2.7.6.2.3.4 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
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>	

Name	Description
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
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
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
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

### 3.2.7.6.2.3.5 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>	
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>	
MESSAGE	Reference to current message object

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
<b>Returning parameter</b>	
NEW_MESSAGE	Reference to current message object

#### i 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 ''.

### 3.2.7.6.2.3.6 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 (from IF_BALI_ITEM_SETTER)	Category of the item Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_FREE_TEXT

### 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: IF_BALI_CONSTANTS=>C_SEVERITY_STATUS
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 ''

Name	Description
<b>Returning parameter</b>	
NEW_FREE_TEXT	Reference to current free text object
Get all free text values:	
GET_ALL_VALUES	
Name	Description
<b>Exporting parameters</b>	
DETAIL_LEVEL	Detail level of the free text Number between '1' and '9' or ''
SEVERITY	Severity of the free text ('Error', 'Warning', etc)
TEXT	Content of the free text

#### i 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 ''.

### 3.2.7.6.2.3.7 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	Category of the item
(from <code>IF_BALI_ITEM_SETTER</code> )	Contains fixed value: <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_EXCEPTION</code>

#### 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
<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
<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 ''
<b>Returning parameter</b>	
NEW_EXCEPTION_OBJ	Reference to current application log exception object

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

#### i 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 ''.

### 3.2.7.6.2.3.8 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
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 is not allowed to delete the log before the expiry date

#### 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

## 3.2.7.6.2.3.9 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
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>	
MESSAGE_TEXT	Message short text of the exception in the logon language

### 3.2.7.6.2.3.10 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

#### 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

### 3.2.7.6.2.3.11 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>• <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_MESSAGE</code>: Item is a message</li><li>• <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_FREE_TEXT</code>: Item is a free text</li><li>• <code>IF_BALI_CONSTANTS=&gt;C_CATEGORY_EXCEPTION</code>: 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 (from <code>IF_BALI_ITEM_GETTER</code> )	UTC time stamp of the item creation

#### 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

### 3.2.7.6.2.3.12 IF\_BALI\_ITEM\_SETTER

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>

## 3.2.7.6.2.3.13 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	Category of the item
(from IF_BALI_ITEM_GETTER)	Contains fixed value: IF_BALI_CONSTANTS=>C_CATEGORY_MESSAGE
LOG_ITEM_NUMBER	Serial number which is the position of the message in the log
(from IF_BALI_ITEM_GETTER)	
SEVERITY	Severity of the message ('Error', 'Warning', etc)
(from IF_BALI_ITEM_GETTER)	
DETAIL_LEVEL	Detail level of the message (number between '1' and '9' or ' ')
(from IF_BALI_ITEM_GETTER)	
TIMESTAMP	UTC time stamp of the message creation
(from IF_BALI_ITEM_GETTER)	
ID	Message ID
NUMBER	Message number
VARIABLE_1	Message variable 1

Name	Description
VARIABLE_2	Message variable 2
VARIABLE_3	Message variable 3
VARIABLE_4	Message variable 4
COUNT	Count of cumulated messages

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

### 3.2.7.7 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 Cloud Platform, cloud connector*.

#### Configuration

The mail server system information must be maintained in a destination of type **MAIL** in the destination service. The following parameters must be defined:

- Host
- Port
- Location ID of the *SAP Cloud Platform, cloud connector*
- User
- Password

Define a communication system in the ABAP Environment pointing to this destination in the destination service. Afterwards, create a communication arrangement for the SAP\_COM\_0548 communication scenario using this communication system.

## Application Programming Interface (API)

Use the `CL_BCS_MAIL_MESSAGE` class to create and send mails. You can specify sender and recipient represented by the `CL_BCS_MAIL_TEXTPART` class. Furthermore, you can add textual or binary attachments represented by the classes `CL_BCS_MAIL_TEXTPART` and `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( 'recipient@yourcompany.com' ).  
    lo_mail->set_subject( 'Test Mail' ).  
    lo_mail->set_main( cl_bcs_mail_txtpart->create_instance(  
        iv_content      = '<h1>Hello</h1><p>This is a test mail.</p>'  
        iv_content_type = 'text/html'  
    ) ).  
    lo_mail->add_attachment( cl_bcs_mail_txtpart->create_instance(  
        iv_content      = 'This is a text attachment'  
        iv_content_type = 'text/plain'  
        iv_filename     = 'Text_Attachment.txt'  
    ) ).  
    lo_mail->send( importing et_status = data(lt_status) ).  
catch cx_bcs_mail into data(lx_mail).  
    "handle exceptions here  
endtry.
```

For more information, see

- [Create Mail Destinations](#)
- [Maintain Communication Systems \[page 1023\]](#)
- [How to Create a Communication Arrangement \[page 1022\]](#)
- [Integrating On-Premise Systems \[page 478\]](#)
- [Configure Access Control \(TCP\)](#)

### 3.2.7.8 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.

### Related Information

[Maintaining Dimensions \[page 581\]](#)

[Maintaining Units of Measurement \[page 587\]](#)

[Conversion Functions for Units of Measurement \[page 595\]](#)

### 3.2.7.8.1 Maintaining Dimensions

Class `CL_UOM_DIM_MAINTENANCE` provides methods for maintaining a dimension.

For more information, see the following:

- [Creating a Dimension \[page 581\]](#)
- [Changing a Dimension \[page 583\]](#)
- [Deleting a Dimension \[page 585\]](#)
- [Reading a Dimension \[page 586\]](#)

#### 3.2.7.8.1.1 Creating a Dimension

Use method `CREATE` to create a new dimension. For customer dimensions, the name of the dimension ID must start with a 'Z'.

### 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

Parameter Name	Field Name	Value Help
	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

### i 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 TYPE REF TO cl_uom_dim_maintenance,
          ls_dim TYPE cl_uom_dim_maintenance=>ty_dim_cre_ts.
    "Get instance
    cl_uom_dim_maintenance=>get_instance(
    RECEIVING
      ro_dimension = lo_dim ).
    ls_dim-dimid = 'ZNEWDI'.
    ls_dim-txdim = 'New Dimension'.
    ls_dim-mass = 89.
  TRY.
    lo_dim->create( EXPORTING dim_cre_ts = ls_dim

```

```

        IMPORTING
            error = DATA(error)
        ).
CATCH cx_uom_error INTO DATA(lo_error).
    out->write( |Exception raised| ).
    out->write( lo_error->get_text( ) ).
ENDTRY.
ENDMETHOD.
ENDCLASS.
```

### 3.2.7.8.1.2 Changing a Dimension

Use method `UPDATE` to change an existing dimension. For customer dimensions, the name of the dimension ID must start with a 'Z' for any changes.

#### 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

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error <b>'X'</b> : Save error

### i 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 TYPE REF TO cl_uom_dim_maintenance,
          ls_dim TYPE cl_uom_dim_maintenance=>ty_dim_upd_ts.
    "Get instance
    cl_uom_dim_maintenance=>get_instance(
    RECEIVING
      ro_dimension = lo_dim ).
    ls_dim-txdim = 'Update Dimension'.
    ls_dim-mass = 88.
    ls_dim-length = 88.
    TRY.
      lo_dim->update( EXPORTING dimid = 'ZNEWDI'
                      dim_upd_ts = ls_dim
                      IMPORTING
                        error = DATA(error)
                      ).
    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( |Exception raised| ).
      out->write( lo_error->get_text( ) ).
    ENDTRY.
  ENDMETHOD.
ENDCLASS.
```

### 3.2.7.8.1.3 Deleting a Dimension

Use method `DELETE` to delete a dimension. For customer dimensions, the name of the dimension ID must start with a 'Z' to allow for deletion.

#### 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

##### i 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 TYPE REF TO cl_uom_dim_maintenance.

    cl_uom_dim_maintenance=>get_instance(
      RECEIVING
      ro_dimension = lo_dim ).

    TRY.
      lo_dim->delete( EXPORTING dimid = 'ZNEWDI'
                      IMPORTING error = DATA(error)
                    ).
    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( |Exception raised| ).
      out->write( lo_error->get_text() ).
```

```
ENDTRY.  
ENDMETHOD.  
ENDCLASS.
```

### 3.2.7.8.1.4 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
	MOLE_QTY	Mole quantity exponent of the dimension

Parameter Name	Field Name	Value Help
	LUMINOSITY	Light exponent of the dimension

### i 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 TYPE REF TO cl_uom_dim_maintenance,
          ls_dim type cl_uom_dim_maintenance=>ty_dim_ts.

    "Get instance
    cl_uom_dim_maintenance=>get_instance(
    RECEIVING
      ro_dimension = lo_dim ).

    try.
      lo_dim->read( exporting dimid = 'AAAADL'
                    importing dim_st = ls_dim
                    ).
    catch cx_uom_error.
    endtry.

    out->write( ls_dim-DIMID ).
    out->write( ls_dim-txdim ).

  ENDMETHOD.

ENDCLASS.

```

## 3.2.7.8.2 Maintaining Units of Measurement

Class CL\_UOM\_MAINTENANCE provides methods for maintaining units of measurement.

For more information, see the following:

- [Creating a Unit of Measurement \[page 588\]](#)
- [Changing a Unit of Measurement \[page 590\]](#)
- [Deleting a Unit of Measurement \[page 592\]](#)

- [Reading a Unit of Measurement \[page 593\]](#)

### 3.2.7.8.2.1 Creating a Unit of Measurement

Use method CREATE to create a unit of measurement. For customer units, the name of the internal unit must start with a 'Z'.

#### 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
	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.

Parameter Name	Field Name	Value Help
	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

### i 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  TYPE REF TO cl_uom_maintenance,
          ls_unit TYPE cl_uom_maintenance=>ty_uom_cre_ts.

    cl_uom_maintenance=>get_instance(
      RECEIVING
      ro_uom = lo_uom ).

    ls_unit-commercial = 'ZYX'.
    ls_unit-technical = 'ZYX'.
    ls_unit-denominator = '1'.
    ls_unit-numerator = '1'.
    ls_unit-dec_disp = '3'.
    ls_unit-long_text = 'Create Unit'.
  TRY.
    lo_uom->create( EXPORTING unit_dimid = 'AAAADL'

```

```

        unit_int      = 'ZYX'
        unit_cre_ts  = ls_unit
IMPORTING error          = DATA(error)
).
CATCH cx_uom_error INTO DATA(lo_error).
  out->write( | Exception raised | ).
  out->write( lo_error->get_text( ) ).
ENDTRY.

ENDMETHOD.
ENDCLASS.
```

### 3.2.7.8.2.2 Changing a Unit of Measurement

Use method `UPDATE` to change a unit of measurement. For customer units, the name of the internal unit must start with a 'Z' for any changes.

#### Import Parameters

Parameter Name	Field Name	Value Help
UNIT		Internal unit of measurement
UNIT_UPD_TS		Structure for updating 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
	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

## Export Parameters

Parameter Name	Value Help
ERROR	<b>Space:</b> No error  <b>'X'</b> : Save error

### i 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  TYPE REF TO cl_uom_maintenance,
          ls_unit  TYPE cl_uom_maintenance=>ty_uom_upd_ts.

    cl_uom_maintenance=>get_instance(
    RECEIVING
      ro_uom = lo_uom ).
```

```

ls_unit-commercial = 'ZYA'.
ls_unit-technical = 'ZYA'.
ls_unit-denominator = '1'.
ls_unit-numerator = '1'.
ls_unit-dec_disp = '5'.
ls_unit-long_text = 'Update Unit'.
ls_unit-text = 'Upd Unit'.

TRY.
    lo_uom->update( EXPORTING unit = 'ZYX'
                      unit_upd_ts = ls_unit
                    IMPORTING
                      error      = DATA(error)
                ).
CATCH cx_uom_error INTO DATA(lo_error).
    out->write( |Exception raised| ).
    out->write( lo_error->get_text( ) ).
ENDTRY.

ENDMETHOD.
ENDCLASS.

```

### 3.2.7.8.2.3 Deleting a Unit of Measurement

Use method `DELETE` to delete a unit of measurement. For customer units, the name of the internal unit must start with a 'Z' to allow for deletion.

#### 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

#### i 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 TYPE REF TO cl_uom_maintenance.

    cl_uom_maintenance=>get_instance(
      RECEIVING
      ro_uom = lo_uom ).

    TRY.
      lo_uom->delete( EXPORTING unit = 'ZYX'
                      IMPORTING error = DATA(error)
                    ).
    CATCH cx_uom_error INTO DATA(lo_error).
      out->write( | Exception raised | ).
      out->write( lo_error->get_text( ) ).
    ENDTRY.

  ENDMETHOD.

ENDCLASS.
```

### 3.2.7.8.2.4 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
	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

### i 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  TYPE REF TO cl_uom_maintenance,
          ls_unit TYPE cl_uom_maintenance=>ty_uom_ts.

    cl_uom_maintenance=>get_instance(
      RECEIVING
      ro_uom = lo_uom ).

    TRY.
      "Unit found
      lo_uom->read( EXPORTING unit = 'ST'
                      IMPORTING unit_st = 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.
```

### 3.2.7.8.3 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 \[page 595\]](#)
- [Determining the SI Unit \[page 598\]](#)
- [Determining the Conversion Factors \[page 599\]](#)

#### 3.2.7.8.3.1 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 ('+') , down ('-' ), commercial ('x') , or no rounding (SPACE).

#### i Note

Make sure that both units are maintained in the measurement unit table and have 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: lo_unit type ref to cl_uom_conversion,
          input  type p decimals 8,
          result type p decimals 8.
    input = '1.98678923'.
    lo_unit = cl_uom_conversion->create( ).
    lo_unit->unit_conversion_simple( exporting input      = input
                                         round_sign = 'X'
                                         unit_in     = 'KG'
                                         unit_out    = 'G'
                                       importing
                                         output = 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( result ).
    ENDIF.
  ENDMETHOD.
```

### 3.2.7.8.3.2 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 type ref to cl_uom_conversion,
          si_unit type cl_uom_conversion=>ty_msehi.
    lo_unit = cl_uom_conversion=>create( ).
    lo_unit->si_unit_get( ).
```

```

EXPORTING
    dimension          = 'MASS'
    unit              = 'G'
IMPORTING
    si_unit           = si_unit
EXCEPTIONS
    dimension_not_found = 1
    unit_not_found     = 2
    si_unit_not_found = 3
    others             = 4
).
IF SY-SUBRC = 0.
    out->write( si_unit ).
ENDIF.
ENDMETHOD.
ENDCLASS.

```

### 3.2.7.8.3.3 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
EXPONENT	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

Parameter Name	Value Help
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      type ref to cl_uom_conversion,
          decimal      type cl_uom_conversion=>ty_andec,
          dimension    type cl_uom_conversion=>ty_dimid,
          numerator    type cl_uom_conversion=>ty_dzaehl,
          denominator  type cl_uom_conversion=>ty_nennr.
    lo_unit = cl_uom_conversion=>create( ).
    lo_unit->unit_parameters_get(
      EXPORTING
        unit           = 'G'
      IMPORTING
        decimals       = decimal
        dimension     = dimension
        numerator     = numerator
        denominator   = denominator
      EXCEPTIONS
        unit_not_found = 1
        others         = 2
    ).
    IF SY-SUBRC = 0.
      out->write( dimension ).
      out->write( decimal ).
      out->write( numerator ).
      out->write( denominator ).
    ENDIF.
  ENDMETHOD.
ENDCLASS.
```

## 3.2.7.9 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 Cloud Platform ABAP environment.

### 3.2.7.9.1 Core Principles

The design of the XCO library is based on the following core principles:

- **Modules**

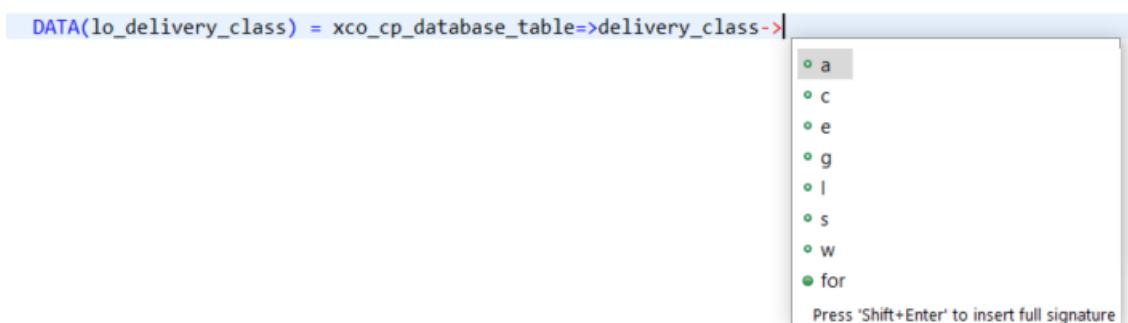
The XCO library is organized as a collection of (largely) independent modules (corresponding to ABAP packages). The public API of each module is exposed via a special class, called the API “class” of the module. It is distinguished from all other XCO classes and interfaces in that it starts with “XCO\_CP\_”. 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 \[page 615\]](#)).

## 3.2.7.9.2 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- Data definition (XCO\_CP\_DATA\_DEFINITION)  
Provides access to enumerations specific to CDS data definitions.
- 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.

- 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 PUT and DELETE operations.
- 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.
- Message (XCO\_CP\_MESSAGE)  
Provides access to enumerations specific to messages, such as the message type.
- Metadata extension (XCO\_CP\_METADATA\_EXTENSION)  
Provides access to enumerations specific to metadata extensions.
- Package (XCO\_CP\_PACKAGE)  
Provides access to enumerations specific to packages.
- 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.
- Service binding (XCO\_CP\_SERVICE\_BINDING)  
Provides access to enumerations specific to service bindings.
- 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.
- Transport (XCO\_CP\_TRANSPORT)  
Provides access to enumerations 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.

### 3.2.7.9.3 Generation

XCO Generation is 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)
- DDLS (Data definitions)

- DDLX (Metadata extensions)
- DEVC (Packages)
- DOMA (Domains)
- DTEL (Data elements)
- INTF (Interfaces)
- MSAG (Message classes)
- SRVD (Service definitions)
- SRVB (Service bindings)
- TABL (Structures and database tables)
- TTYP (Table types)

For each supported object type two kinds of operations are provided:

- PUT Create or update the object according to a provided specification
- DELETE Delete the object if it exists

### 3.2.7.9.3.1 Design of the XCO Generation APIs

As with the whole XCO library, the design of the XCO Generation APIs is governed by core principles:

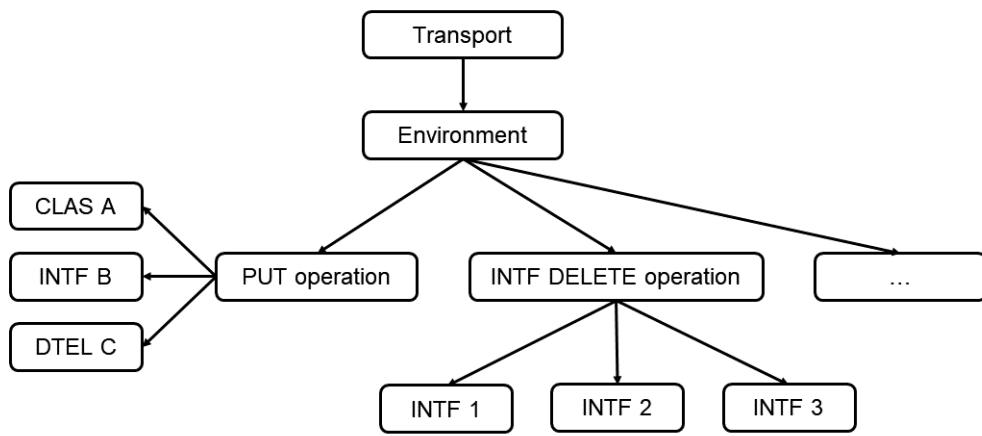
- **Environment**

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 non-package objects. 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.

#### *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.

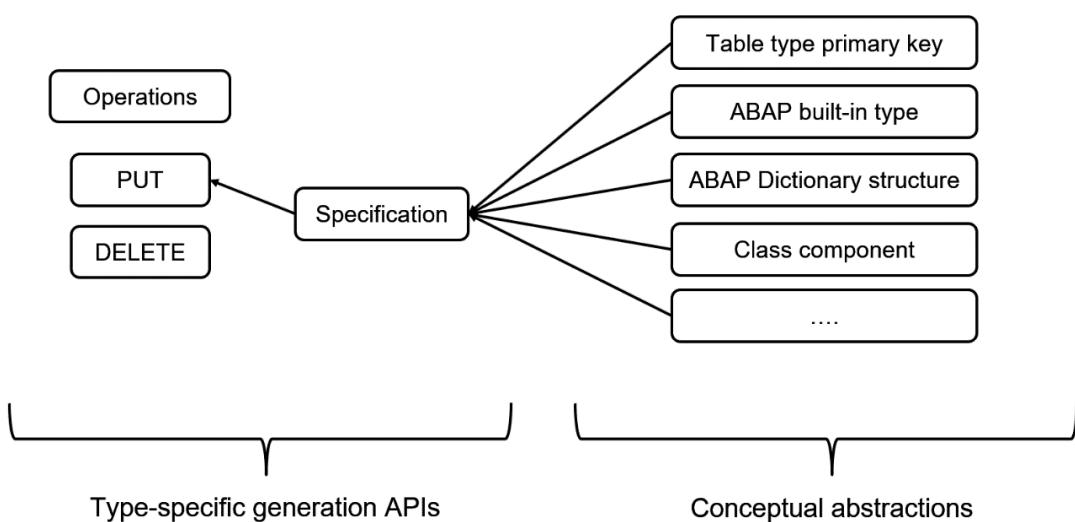
#### *Errors during operation execution*

If the execution of an operation fails, a CX\_XCO\_GEN\_PUT\_EXCEPTION (resp. CX\_XCO\_GEN\_DELETE\_EXCEPTION) is raised for a PUT (resp. DELETE) operation. The 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' ).
```

## 3.2.7.9.4 ABAP Repository

The XCO ABAP Repository module provides APIs that allow to easily navigate through the ABAP Repository and access the content for objects of selected object types.

## Query APIs

The XCO ABAP Repository Query APIs provide an object selection functionality 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->table->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.
```

The XCO ABAP Repository APIs can also be used to programmatically release development objects, e.g. data elements:

#### « 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  
).
```

## Read APIs

The XCO ABAP Repository Read APIs allow to access the content for objects of the following types:

- APLO (Application log object)
- DDLS (Data Definition)
- DOMA (Domain)
- DTEL (Data element)
- MSAG (Message class)
- TABL (Structure and database table)
- TTYP (Table type)

Where applicable the version of the content to be read can be specified, e.g. when accessing the content of objects of the ABAP Dictionary it can be selected whether the newest or the active version shall be considered.

All Read APIs are strongly typed and tailored to the specific characteristics of the object type. For example, reading the fields of a database table and performing a “drill-down” to detect which fields are backed by a data element and furthermore which of those are backed by a domain can be accomplished like this:

#### « Sample Code

```
DATA(lo_database_table) = xco_cp_abap_repository=>object->tabl-  
>database_table->for( 'ZMY_DBT' ).  
LOOP AT lo_database_table->fields->all->get( ) INTO DATA(lo_field).  
    DATA(lo_field_type) = lo_field-  
>content( xco_cp_abap_dictionary=>object_read_state->active_version  
        )->get_type( ).  
    IF lo_field_type->is_data_element( ) EQ abap_true.  
        DATA(lo_data_element_data_type) = lo_field_type->get_data_element( )  
            ->content( )  
            ->get_data_type( ).  
        IF lo_data_element_data_type->is_domain( ) EQ abap_true.  
            DATA(lo_domain_format) = lo_data_element_data_type->get_domain( )  
                ->content( xco_cp_abap_dictionary=>object_read_state->newest_version  
                    )->get_format( ).  
            " LO_BUILT_IN_TYPE represents the built-in type of the ABAP Dictionary  
            that  
            " is used for the format of the domain, e.g. CHAR 10.  
            DATA(lo_builtin_type) = lo_domain_format->get_builtin_type( ).  
        ENDIF.  
    ENDIF.
```

```
ENDLOOP.
```

### 3.2.7.9.5 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.

#### 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 symmsg(
    msgty = 'W'
    msgid = 'MSG_CLASS'
    msgno = '001'
    msgv1 = '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_symmsg       = 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( ).
```

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.

### 3.2.7.9.6 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.

#### Integration with the ABAP Repository APIs

An important differentiation to be aware of when using the XCO ABAP Repository APIs to locate CDS objects is that between DDLS and STOB, both of which define object types in the ABAP Repository and are accessible via XCO\_CP\_ABAP\_REPOSITORY=>OBJECTS.

A DDLS object represents the source that is used to define a CDS entity which is represented by a STOB object. By convention, the name of the STOB object defined inside a DDLS object equals the name of the DDLS object.

As a C1-release for a CDS entity always pertains to the STOB object contained inside a DDLS object only the STOB object of a C1-released CDS entity will be visible via XCO\_CP\_ABAP\_REPOSITORY. The corresponding DDLS object will be treated as invisible.

## 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().  
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\_CP\_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

## 3.2.7.9.7 Correction and Transport System

The XCO CTS module provides APIs which support the creation and release of Workbench 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 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( ).
```

## 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.

## 3.2.7.9.8 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 )
```

### 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 \[page 617\]](#)).

## Related Information

[ADT \[page 617\]](#)

[JSON \[page 618\]](#)

[Regular Expression \[page 620\]](#)

[String \[page 620\]](#)

[UUID \[page 622\]](#)

### 3.2.7.9.8.1 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).

### 3.2.7.9.8.2 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":"7cd44ffff-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 = 7cd44ffff-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\_CP\_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( ).
```

### 3.2.7.9.8.3 Regular Expression

Regular expressions can be used to match strings against patterns or extract substrings that match certain criteria.

Independent of XCO, CL\_ABAP\_REGEX provides common regular expression standards for working with regular expressions in ABAP. 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

```
DATA(lo_posix_engine) = xco_cp_regular_expression=>engine->posix(
    iv_ignore_case = abap_true
).
DATA(lo_pcrc_engine) = xco_cp_regular_expression=>engine->pcrc(
    iv_ignore_case      = abap_false
    iv_enable_multiline = abap_false
).
```

The parameters of the POSIX and PCRE 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 ].
```

### 3.2.7.9.8.4 String

Besides providing a straightforward integration with regular expressions (see [Regular Expression \[page 620\]](#)), 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.

```

### 3.2.7.9.8.5 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 ) ).
```

### 3.2.7.9.9 Code Samples

#### 3.2.7.9.9.1 Generation

##### 3.2.7.9.9.1.1 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_clas_name       TYPE sxco_ad_object_name VALUE
'ZXCO_CP_BP_VACATION_REQUEST',co_srwd_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_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_srwd
    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_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-dtels->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-tables-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->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->dats ).  

  lo_specification->add_field( 'END_DATE' )->set_type( xco_cp_abap_dictionary=>built_in_type->dats ).  

  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_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_srvd.
    DATA(lo_specification) = io_put_operation->for-srvd-
>add_object( co_srvd_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_srvb.
    DATA(lo_put_operation) = mo_environment->for-srvb-
>create_put_operation( ).
    DATA(lo_specification) = lo_put_operation->add_object( co_srvb_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_srvd_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
"!
"!
"! 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_clas_name       TYPE sxco_ad_object_name VALUE
'ZXCO_CP_BP_VACATION_REQUEST', co_srwd_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:
  delete_srvb,
  delete_srwd,
  delete_clas,
  delete_bdef,
  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_ddlx( ).
  delete_ddls( ).
  delete_tabl_dbt( ).
  delete_dtel( ).
  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_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.
```

### 3.2.7.9.9.1.2 Generation Error Handling

Any errors that occur during the execution of a PUT or DELETE operation are communicated via exceptions of types CX\_XCO\_GEN\_PUT\_EXCEPTION and CX\_XCO\_GEN\_DELETE\_EXCEPTION. 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.

```

## 3.2.8 UI Development

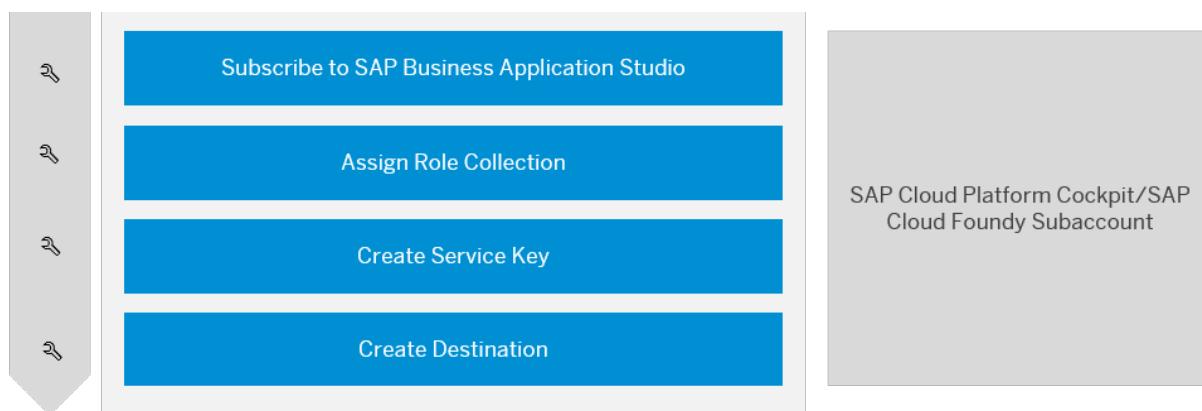
### 3.2.8.1 Deploy an SAP Fiori Application UI to ABAP Using SAP Business Application Studio

Get an overview about how to create and deploy an SAP Fiori application to ABAP using SAP Business Application Studio.

#### 1. Setting Up SAP Business Application Studio

##### i Note

- You must be a member of a global account. See [Managing Global Accounts and Subaccounts Using the Cockpit \[page 754\]](#).
- You must be an organization manager in your Cloud Foundry subaccount. See [Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#).
- You must be a security administrator in your Cloud Foundry subaccount. See [Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#).
- You have created an ABAP service instance. See [Creating an ABAP System](#).



##### Legend

Task for Administrators

1. To create and deploy SAP Fiori application UIs, you, as an administrator in a Cloud Foundry subaccount, must first subscribe to SAP Business Application Studio. See [Subscribe to SAP Business Application Studio](#) and [Set Up SAP Business Application Studio for Development](#) for an in-depth tutorial.
2. Now you have to add users to your role collection and assign permissions for SAP Business Application Studio. See [Add Users to Role Collections \[page 817\]](#) and [Manage Authorizations](#).

##### i Note

The administrator and developer role collections  
(Business\_Application\_Studio\_Administrator,

`Business_Application_Studio_Developer`, and `Business_Application_Studio_Extension_Deployer`) are created automatically upon subscription.

3. Continue with the creation of a service key for your ABAP system. See [Creating a Service Key for the Destination Service Instance \(Optional\)](#).

**i Note**

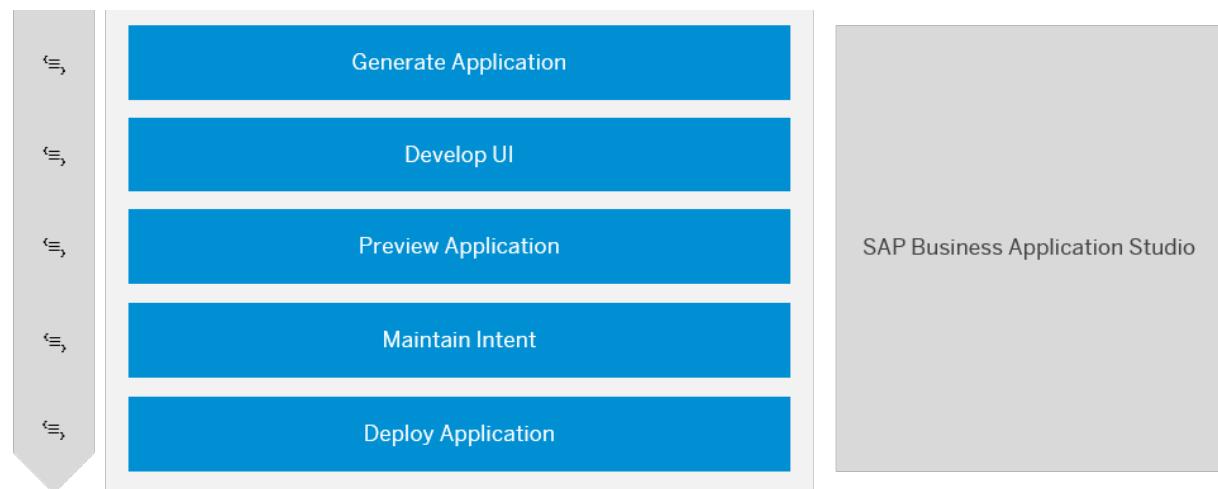
You must have already set up your ABAP system. See [Creating an ABAP System](#).

4. Set up a destination in the SAP Business Application Studio subaccount. See [Creating a Destination to the ABAP System for SAP Business Application Studio \(Optional\)](#).

## 2. Generating and Deploying Your Application

**i Note**

- You have created and cloned a software component. See [How to Create Software Components \[page 1073\]](#) and [How to Clone Software Components \[page 1075\]](#).
- You have a development package in ABAP Development Tools for Eclipse. See [Creating ABAP Packages](#).
- You have created a RAP business object with a service binding of type OData version 2.0 (ODATA V2/ UI).
- You have an open transport request.



**Legend**

Task for Developers

1. Start with the creation of your application and connect it to your SAP system. See [Generate an Application](#).
2. Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).

- After you have successfully generated your application and developed the UI, you can preview the application. See [Previewing Application](#).
- Add a semantic object and action to the `manifest.json` file. This step is necessary for the navigation of your app so that it can be added as a tile to the launchpad.

#### **! Restriction**

- The UIAD object is created upon deployment only if the intent is defined in the manifest
- Only one intent is supported
- If you change the ID of the intent, the existing SAP Fiori launchpad content content will break

- Now you can deploy your application with command `npm run deploy`. See [Deploying an Application](#).

#### **i Note**

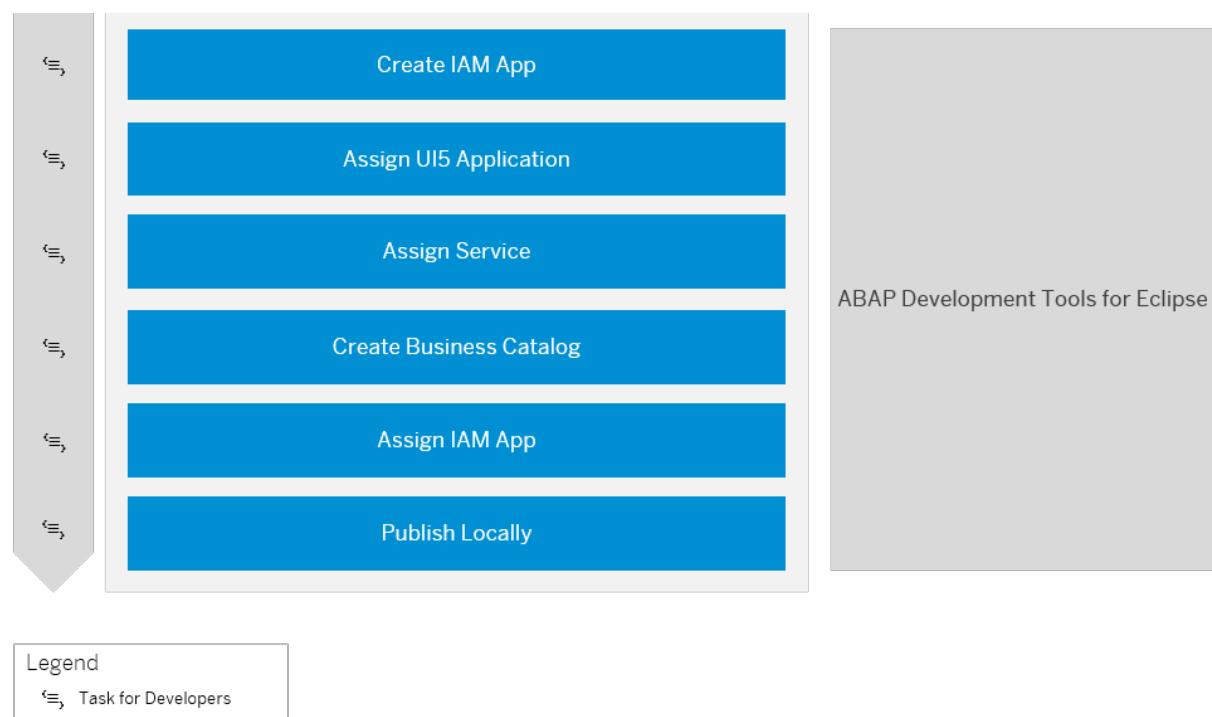
The following objects are created:

- WAPA
- UIAD
- SICF
- SMIM

#### **→ Remember**

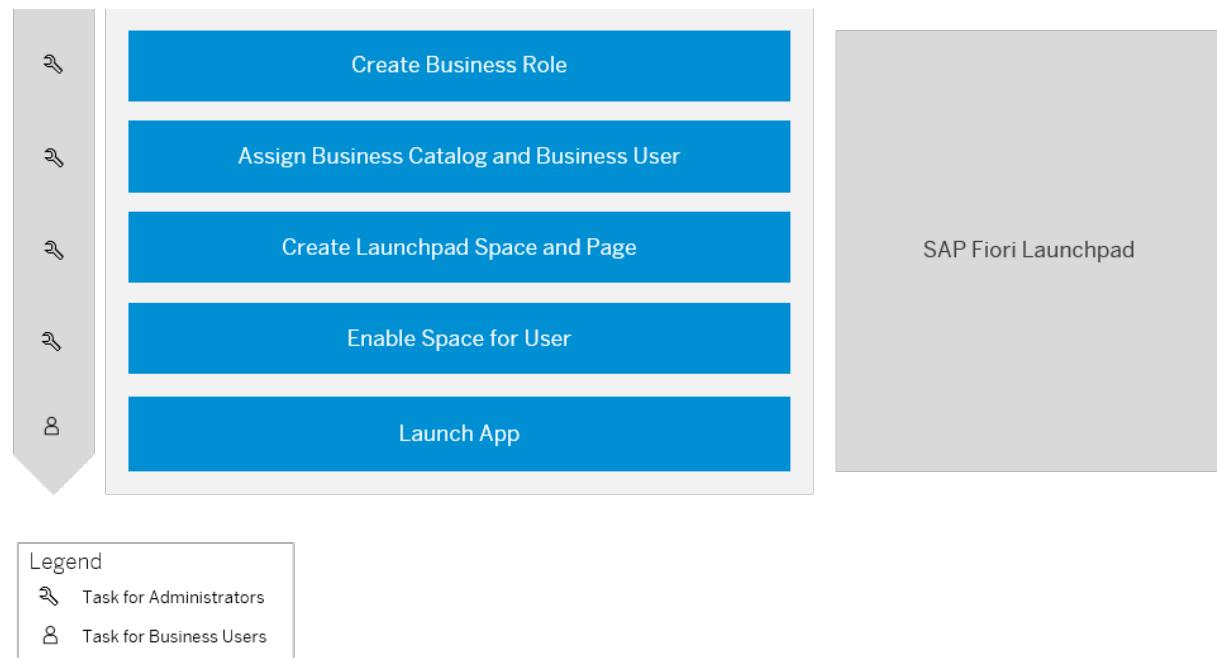
Business catalog `SAP_A4C_BC_DEV_UID_PC` must be assigned to your ABAP user.

## 3. Creating and Publishing Your IAM App



1. In ABAP Development Tools for Eclipse, you have to log on as a developer to create an Identity and Access Management (IAM) application, assign a UI5 application and a service, and maintain authorizations (steps 1-3 in the figure above). See .
2. Once you have created your app, you have to create a business catalog. See .
3. Assign your IAM app.
4. Publish it locally.

## 4. Setting Up SAP Fiori Launchpad to Launch Your App



1. In the SAP Fiori launchpad, create a business role with the Maintain Business Roles app. See [Maintain Business Roles \[page 1043\]](#).
2. Assign a business catalog and business user to the business role. See [Maintain Business Users \[page 1038\]](#).
3. Create a launchpad space and page. See [How to Create a Space and Page for a Business Role](#).

**i Note**

To use this app, your user must be assigned to business catalog SAP\_CORE\_BC\_UI\_FLD.

4. 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](#).
5. Now you can launch your app.

## Related Information

[SAP Business Application Studio](#)

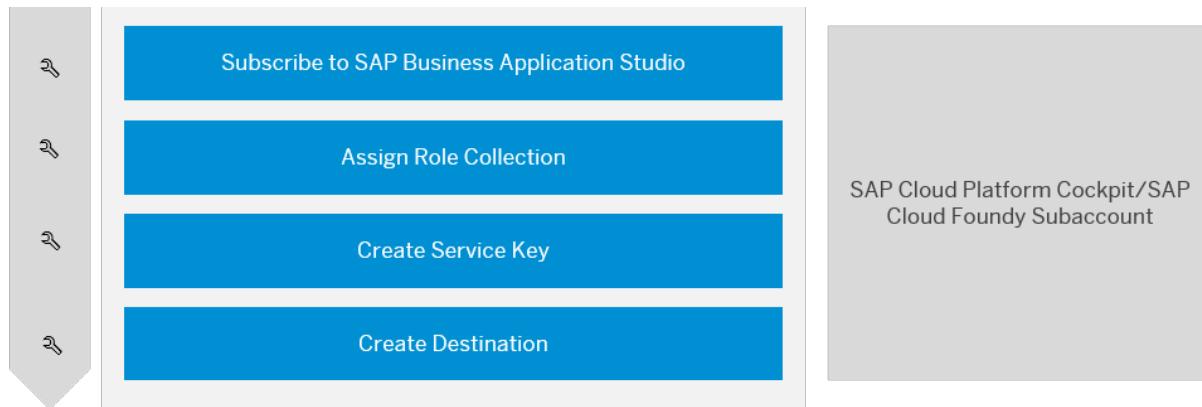
### 3.2.8.2 Deploy an SAP Fiori Application UI to Cloud Foundry Using SAP Business Application Studio

Get an overview about how to create and deploy an SAP Fiori application to Cloud Foundry using SAP Business Application Studio.

#### 1. Setting Up SAP Business Application Studio

##### i Note

- You must be a member of a global account. See [Managing Global Accounts and Subaccounts Using the Cockpit \[page 754\]](#).
- You must be an organization manager in your Cloud Foundry subaccount. See [Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#).
- You must be a security administrator in your Cloud Foundry subaccount. See [Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#).
- You have created an ABAP service instance. See [Creating an ABAP System](#).



##### Legend

Task for Administrators

1. To create and deploy SAP Fiori application UIs, you, as an administrator in a Cloud Foundry subaccount, must first subscribe to SAP Business Application Studio. See [Subscribe to SAP Business Application Studio](#) and [Set Up SAP Business Application Studio for Development](#) for an in-depth tutorial.
2. Now you have to add users to your role collection and assign permissions for SAP Business Application Studio. See [Add Users to Role Collections \[page 817\]](#) and [Manage Authorizations](#).

### i Note

The administrator and developer role collections (`Business_Application_Studio_Administrator`, `Business_Application_Studio_Developer`, and `Business_Application_Studio_Extension_Deployer`) are created automatically upon subscription.

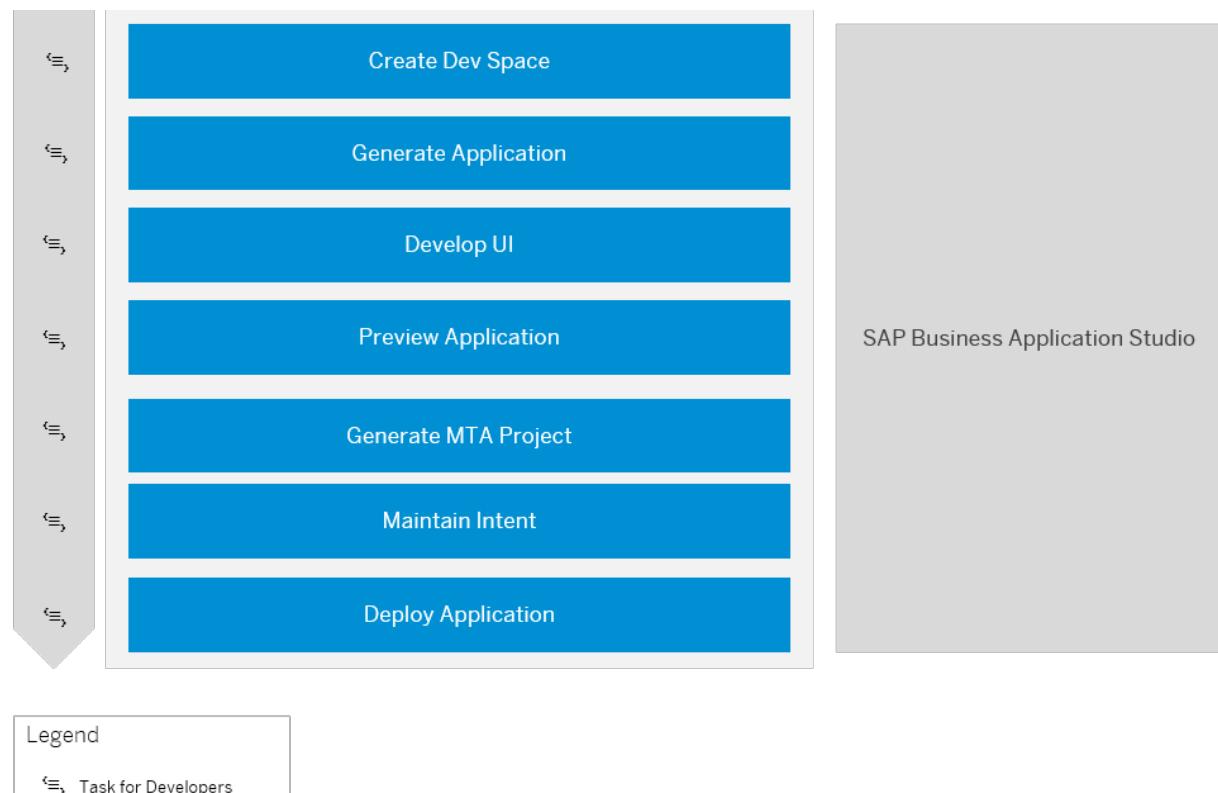
3. Continue with the creation of a service key for your ABAP system. See [Creating a Service Key for the Destination Service Instance \(Optional\)](#).

### i Note

You must have already set up your ABAP system. See [Creating an ABAP System](#).

4. Set up a destination in the SAP Business Application Studio subaccount. See [Creating a Destination to the ABAP System for SAP Business Application Studio \(Optional\)](#).

## 2. Generating and Deploying Your Application



1. Prepare your development environment by creating a dev space. See [Requirements for SAP Business Application Studio](#) and [Create a Dev Space for SAP Fiori Apps](#) for an in-depth tutorial.

### → Tip

Make sure to log on to your Cloud Foundry organization and space before continuing with the next step.

2. Now it's time to generate your Fiori elements application and connect to your SAP system. See [Generate an Application](#).
3. Continue with the development of the UI, for example, with the help of guided development. See [Implement Features using Guided Development](#).
4. After you have successfully generated your application and developed the UI, you can preview the application. See [Previewing Application](#).
5. Continue with the generation of an MTA project by executing command `npx fiori add deploy-config`.
6. Add a semantic object and action to the `manifest.json` file. This step is necessary for the navigation of your app so that it can be added as a tile to the launchpad. See [Add Content to the Launchpad](#).
7. Now you can deploy your application with command `npm run deploy`. See [Deploying an Application](#).

## Related Information

[SAP Business Application Studio](#)  
[SAP Fiori Tools](#)  
[SAP Fiori Overview](#)  
[SAP Cloud Platform Portal](#)

## 3.3 Development in the Kyma Environment

The Kyma environment allows you to extend existing SAP systems with your own Functions or microservices.

The user-friendly Kyma Console UI integrated with SAP Cloud Platform allows you to easily create instances of external services. This way, you can use the functionality they provide to build your applications. If you prefer working with command-line tools, you can use the Kubernetes command-line tool, [kubectl](#) ↗, to set up service-application communication and create Functions.

### [Using Services in the Kyma Environment \[page 637\]](#)

Kyma environment allows you to extend the SAP and non-SAP services to build and deploy your own applications.

### [Create Functions \[page 646\]](#)

Access the Kyma environment and start creating extensions for SAP systems using Functions.

### 3.3.1 Using Services in the Kyma Environment

Kyma environment allows you to extend the SAP and non-SAP services to build and deploy your own applications.

To use functionality provided by external services in your applications, you must perform certain steps. To do so, use either the Kyma Console UI or kubectl.

Description	Link
1. Create an instance of a service available in the Service Catalog. This catalog lists services you are entitled to use in your subaccount and services provided by the cloud providers, such as AWS, Azure, or GCP.	<a href="#">Creating Service Instances [page 638]</a>
2. Bind a service instance to the application.	<a href="#">Binding Service Instances to Applications [page 642]</a>
3. Create credentials that allow your application to communicate with the service.	<a href="#">Creating Credentials [page 644]</a>

→ Tip

Read more about the provisioning and binding flow in Kyma .

### 3.3.1.1 Creating Service Instances

Create instances of services and use them to extend your own applications.

#### [Create Service Instances Using the Kyma Console UI \[page 638\]](#)

Create instances of services available in the Kyma environment.

#### [Create Service Instances Using kubectl \[page 639\]](#)

You can use the Kubernetes command-line tool, **kubectl**, to create your service instances.

#### [Register Cloud Providers \[page 641\]](#)

Kyma environment allows you to register cloud providers, such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP), to extend the Service Catalog with additional services.

### 3.3.1.1.1 Create Service Instances Using the Kyma Console UI

Create instances of services available in the Kyma environment.

## Prerequisites

Before you proceed, make sure you addressed the prerequisites listed in the table:

Action	Mandatory	Link
Enable the Kyma environment.	Yes	<a href="#">Provision Kyma Environment [page 1083]</a>

Action	Mandatory	Link
Ensure you are entitled to provision and consume a service using the Service Catalog.	Yes	<a href="#">Configure Entitlements and Quotas for Subaccounts [page 781]</a>
Register cloud providers if you want to use the services they provide.	No	<a href="#">Register Cloud Providers [page 641]</a>

## Context

Follow the steps to create a service instance:

## Procedure

1. In the Kyma Console UI, go to {YOUR\_NAMESPACE} > Service Management > Catalog.
2. Search for the service you want to instantiate.
3. Click **+ Add** and provide the following details:
  - **Name** - a unique name for your service instance. If you do not provide any custom name, the system will automatically generate one.
  - **Plan** - a consumption plan for your service instance. The service plan is the representation of the costs and benefits for a given variant of a particular service.
  - **Parameters** - click **Add parameters** to provide specific parameters for your plan in the JSON format.

→ Tip

For details on available service plans and parameters, see [Supported Service Plans for SAP S/4HANA Cloud \[page 691\]](#) and [Communication Arrangement JSON File - Properties \[page 667\]](#).

4. Click **Create**.

Wait for the service instance to have the status **RUNNING**. You can now bind your application to the service instance and create credentials to access it.

### 3.3.1.1.2 Create Service Instances Using kubectl

You can use the Kubernetes command-line tool, **kubectl**, to create your service instances.

## Prerequisites

Before you proceed, make sure you addressed the prerequisites listed in the table:

Action	Mandatory	Link / Remarks
Install and set up <b>kubectl</b> .	Yes	<a href="#">Install kubectl</a> ↗
Download the <b>kubeconfig</b> file to access the cluster using the command line.	Yes	You can find the kubeconfig file under <a href="#">Settings</a> ➤ <a href="#">General Settings</a> ➤ .
Enable the Kyma environment.	Yes	<a href="#">Provision Kyma Environment [page 1083]</a>
Register cloud providers if you want to use the services they provide.	No	<a href="#">Register Cloud Providers [page 641]</a>

## Context

Open the terminal and follow the steps to create a service instance:

## Procedure

1. If you do not have a Namespace already, create one:

```
kubectl create namespace {YOUR_NAMESPACE}
```

2. Create a service instance:

### i Note

To create a proper service instance, replace the variables with actual values.

```
cat <<EOF | kubectl apply -f -
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceInstance
metadata:
  name: {INSTANCE_NAME}
  namespace: {YOUR_NAMESPACE}
spec:
  ServiceClassExternalName: {SERVICE_NAME}
  ServicePlanExternalName: {PLAN_NAME}
  parameters:
    param-1: value-1
    param-2: value-2
EOF
```

### → Tip

To list the services available in your Namespace, run:

```
kubectl get serviceclasses -n {YOUR_NAMESPACE}
```

To view the details of the external service, run:

```
kubectl get serviceclasses -n {YOUR_NAMESPACE} {SERVICE_NAME} -o yaml
```

To list service plans available in your Namespace, run:

```
kubectl get serviceplans -n {YOUR_NAMESPACE}
```

To view the details of a particular service plan, run:

```
kubectl get serviceplans -n {YOUR_NAMESPACE} {SERVICE_NAME} -o yaml
```

## Next Steps

Once your service instance is up and running, you can bind it to the application. See [Bind Service Instances to Applications Using kubectl \[page 643\]](#) for details.

### 3.3.1.1.3 Register Cloud Providers

Kyma environment allows you to register cloud providers, such as Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP), to extend the Service Catalog with additional services.

#### Prerequisites

- Enable the Kyma environment. For details, see [Provision Kyma Environment \[page 1083\]](#).

#### Context

Follow the steps to register cloud providers:

#### Procedure

1. In the Kyma Console UI, go to  > [Service Management](#) > [Catalog](#) > [Add-Ons](#).
2. Select the tile with the cloud provider you would like to register.
3. Click [+ Add once](#) and fill in the fields using the instructions available in the [Documentation](#) section.

### 3.3.1.2 Binding Service Instances to Applications

Bind the service instances to your applications so that they can communicate with one another.

#### [Bind Service Instances to Applications Using the Kyma Console UI \[page 642\]](#)

You can bind the service instance to any workload running in the Kyma runtime, such as a Function or a microservice.

#### [Bind Service Instances to Applications Using kubectl \[page 643\]](#)

As an alternative to the Kyma dashboard, you can use **kubectl** to bind your service to the application and establish a connection between the two.

#### 3.3.1.2.1 Bind Service Instances to Applications Using the Kyma Console UI

You can bind the service instance to any workload running in the Kyma runtime, such as a Function or a microservice.

#### Context

Follow the steps to bind the application to a service instance:

#### Procedure

1. Go to [{YOUR\\_NAMESPACE}](#) [Service Management](#) [Instances](#) [Bound Applications](#) and select the instance from the list.
2. Click [Bind Application](#).
3. Use the drop-down menu to provide the following information:
  - a. [Select Application](#) - select the already existing application. For your convenience, applications are already grouped by kind.
  - b. [Set prefix for injected variables](#) - use a prefix if your Secret contains fields with generic names. This way, when you want to reuse the Secret, the fields will already have unique names.
  - c. [Select existing credentials](#) - if you already have a set of credentials in place, you can add them to your binding. For details, see [Creating Credentials \[page 644\]](#).
4. Confirm with [Bind Application](#).

You can see the bound application on the list. Go to the [Credentials](#) tab to view the Secret resource.

### 3.3.1.2.2 Bind Service Instances to Applications Using kubectl

As an alternative to the Kyma dashboard, you can use **kubectl** to bind your service to the application and establish a connection between the two.

#### Prerequisites

To successfully bind your application to the service instance, you first need to create the credentials the binding uses to access the service. For details, see [Create Credentials Using kubectl \[page 645\]](#).

#### Context

To enable the connection between the application and the service instance, manually create a ServiceBindingUsage custom resource that is a link between the service instance and the application that you create.

#### Procedure

Create a ServiceBindingUsage resource:

**i** Note

To create a proper service instance, replace the variables with actual values.

```
cat <<EOF | kubectl apply -f -
apiVersion: servicecatalog.kyma-project.io/v1alpha1
kind: ServiceBindingUsage
metadata:
  name: {SERVICE_BINDING_USAGE_NAME}
  namespace: {NAMESPACE_NAME}
spec:
  serviceBindingRef:
    name: {SERVICE_BINDING_NAME}
  usedBy:
    kind: deployment
    name: {APPLICATION_NAME}
parameters:
  envPrefix:
    name: "PREFIX_"
EOF
```

Applying such a resource allows the system to inject the Secret created along with the ServiceBinding to the application. To inspect your ServiceBindingUsage resource, run:

```
kubectl get servicebindingusage -n {YOUR_NAMESPACE} {SERVICE_BINDING_USAGE_NAME}
-o yaml
```

### 3.3.1.3 Creating Credentials

Create service credentials your application will use to call the service and retrieve information from it.

The application needs information necessary to connect to the service and authenticate it. The Service Catalog uses the ServiceBinding custom resource to create the Kubernetes Secret object which contains the credentials necessary to call the service.

#### [Create Credentials Using the Kyma Console UI \[page 644\]](#)

To use a given service, you need credentials to access it.

#### [Create Credentials Using kubectl \[page 645\]](#)

To allow your application to call the service, you need specific credentials.

#### 3.3.1.3.1 Create Credentials Using the Kyma Console UI

To use a given service, you need credentials to access it.

#### Context

When you use the Kyma dashboard to bind a service instance to the application, the system **automatically** creates a [ServiceBindingUsage](#) Kyma custom resource under the hood, along with the Kubernetes Secret resource with credentials necessary to access the service.

You can add multiple Secrets that you can use when binding your service instances to applications. Follow the steps to create additional credentials for your service:

#### Procedure

1. In the Kyma Console UI, go to {YOUR\_NAMESPACE} > Service Management > Instances > Credentials
2. Click [Create Credentials](#).

An entry with credentials is added for the service instance. Click [Secret](#) to view the credentials.

#### Note

The parameters provided in the Secret may differ depending on the service instance you create it for.

### 3.3.1.3.2 Create Credentials Using kubectl

To allow your application to call the service, you need specific credentials.

#### Context

To ensure your application can call the service instance, you must create a ServiceBinding resource. This resource is a link between a service instance and an application that you create to request credentials or configuration details for a given service.

#### Procedure

1. Create a ServiceBinding resource :

**i Note**

To create a proper service instance, replace the variables with actual values.

```
cat <<EOF | kubectl apply -f -
apiVersion: servicecatalog.k8s.io/v1beta1
kind: ServiceBinding
metadata:
  name: {SERVICE_BINDING_NAME}
  namespace: {YOUR_NAMESPACE}
spec:
  instanceRef:
    name: {SERVICE_INSTANCE_NAME}
EOF
```

Applying such a resource allows the system to create credentials in the form of a Kubernetes Secret resource. To inspect your ServiceBinding resource, run:

```
kubectl get servicebindings -n {YOUR_NAMESPACE} {SERVICE_INSTANCE_NAME} -o
yaml
```

2. To check if the Secret containing the service credentials is available in your Namespace, run:

```
kubectl get secrets -n {YOUR_NAMESPACE}
```

The result should be similar to the following:

NAME	TYPE	DATA	AGE
{SERVICE_BINDING_NAME}	Opaque		
91s			8

**i Note**

To use the credentials, make sure your application is bound to the service instance. For details, see [Bind Service Instances to Applications Using kubectl \[page 643\]](#).

## 3.3.2 Create Functions

Access the Kyma environment and start creating extensions for SAP systems using Functions.

### Overview

You can extend a given SAP system in the Kyma environment with the so-called [Functions](#) that are based on the Function custom resource. Functions are simple code snippets that you can run without provisioning or managing servers. They implement the exact business logic you define in their code.

Follow these tutorials to get familiar with Functions and learn how to use them.

#### i Note

All tutorials contain steps that you can perform both on the UI ([Console UI](#) tab) and the terminal after downloading the kubeconfig file with the cluster configuration ([CLI](#) tab).

Description	Link
Choose a Namespace in your cluster and create a simple Function that returns the "Hello World!" message.	<a href="#">Create a Function</a>
Expose a sample Function outside the cluster, to an unsecured endpoint that accepts all request methods.	<a href="#">Expose a Function with an API Rule</a>
Extend your Function by binding it to an instance of a sample Redis service. As a result, you will receive the Function with encoded Secrets to the service that you can later use to implement more advanced business logic for your SAP systems.	<a href="#">Bind a Service Instance to a Function</a>
Configure your Function to react to the selected events sent by an external system.	<a href="#">Trigger a Function with an event</a>

#### ⚠ Caution

Make sure that the Function is error-free. For example, it cannot include any syntactic errors or return an intentional HTTP error, as this may lead to loss of event data. If the Function does not respond, Kyma will retry to reach it for about 5 minutes. After this time, the event will be discarded.

# 4 Extensions

SAP Cloud Platform is the extension platform from SAP. It 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

All standard SAP solutions are offered with customizing capabilities. Additionally, customers often have their own requirements for innovative or industry-specific extensions and the SAP Cloud Platform extension capability can help them build, deploy, and operate their new functionalities easily and securely.

You can use the SAP Cloud Platform to 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.

In SAP Cloud Platform, you have these options to extend your SAP solution:

- Extensions with automated configurations in the Cloud Foundry and Kyma environments: applicable for SAP S/4HANA Cloud, and SAP SuccessFactors.
- Extensions with automated configurations in the Kyma environment: applicable for SAP S/4HANA Cloud, SAP SuccessFactors, SAP Cloud for Customer, SAP Commerce Cloud, and SAP Field Service Management.
- Extensions with manual configurations in the Cloud Foundry environment: applicable for SAP S/4HANA Cloud, SAP SuccessFactors, and SAP Cloud for Customer.

### Extensions with Automated Configurations

SAP Cloud Platform provides 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.

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 SAP Cloud Platform in a single formation in the SAP Cloud Platform Cockpit. See [Assigning SAP Systems to a Formation \[page 652\]](#).

The following SAP solutions currently support the automated configurations:

- SAP S/4HANA Cloud
- SAP SuccessFactors
- SAP Cloud for Customer
- SAP Commerce Cloud
- SAP Field Service Management

## Extensions with Manual Configurations

If the automated integration with SAP Cloud Platform does not support your scenario, you can configure the integration manually. Using manual configurations, you can extend these SAP solutions on SAP Cloud Platform, Cloud Foundry environment:

- SAP S/4HANA Cloud
- SAP SuccessFactors
- SAP Cloud for Customer

## Troubleshooting

To get interactive support to help troubleshoot issues related to extensions, see [Guided Answers for Extensions](#)



## Related Information

[Extending SAP Solutions Using Automated Configurations \[Feature Set A\] \[page 648\]](#)

[Extending SAP Solutions Using Manual Configurations \[page 709\]](#)

[Guided Answers for Extensions](#)



## 4.1 Extending SAP Solutions Using Automated Configurations [Feature Set A]

### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

SAP Cloud Platform is the go-to cloud-native extensibility platform for the Intelligent Enterprise. It offers a standard way for extending SAP solutions and developing event-driven extensions and applications.

## Overview

SAP Cloud Platform provides 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 SAP Cloud Platform 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. The following SAP system types are supported:

- SAP S/4HANA Cloud
- SAP SuccessFactors
- SAP Commerce Cloud
- SAP Cloud for Customer
- SAP Field Service Management

### i Note

The content in this section is only relevant for cloud management tools feature set A. See [Cloud Management Tools - Feature Set Overview](#).

### i Note

To enable the integration between your SAP solution and SAP Cloud Platform and to build extension applications, you need to:

1. Connect the corresponding SAP system with the SAP Cloud Platform global account.  
During the pairing process you create an integration token which is then used by the SAP system administrator to configure the integration on the SAP system side. See [Registering an SAP System \[page 650\]](#).
2. Make the SAP system accessible in the SAP Cloud Platform subaccounts in which you want to build your extension applications.  
To do so, you create a formation containing one or more different systems assigned to a common subaccount. See [Assigning SAP Systems to a Formation \[page 652\]](#).  
For the systems of type SAP Commerce Cloud, SAP Cloud for Customer, and SAP Field Service Management, you can continue with developing your extension application.  
For systems of type SAP S/4HANA 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.
3. For systems of type SAP S/4HANA Cloud and SAP SuccessFactors, configure the communication flow for the extension application.  
To be able to consume the SAP S/4HANA 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 SAP Cloud Platform 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 \[page 654\]](#)

[Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 692\]](#)

[Extending SAP Customer Experience Products in the Kyma Environment \[page 705\]](#)

### 4.1.1 Registering an SAP System

To connect an SAP solution system with an SAP Cloud Platform global account, you first need to register the system.

#### Prerequisites

##### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

You are an administrator of the global account where you want to register your SAP system.

#### Context

The registration process is based on an integration token that is used for the pairing of the system and the corresponding SAP Cloud Platform global account. You create the token in the SAP Cloud Platform cockpit, and then use it to configure the integration on the corresponding SAP solution side.

The following SAP system types are supported:

- SAP S/4HANA Cloud (available for Cloud Foundry and Kyma environment)
- SAP SuccessFactors (available for Cloud Foundry and Kyma environment)
- SAP Commerce Cloud (available for Kyma environment)
- SAP Cloud for Customer (available for Kyma environment)
- SAP Field Service Management (available for Kyma environment)

The registration process has the following states displayed in the SAP Cloud Platform cockpit:

- Pending* - the integration 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 integration 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 SAP Cloud Platform cockpit.
- *Failed* - the registration has failed.

## Procedure

1. In the SAP Cloud Platform cockpit, navigate to your global account, and then choose ► *System Landscape* ➤ *Systems* from the left hand-side navigation.
2. In the *Systems* panel, choose *Register System*.
3. In the *Register System* dialog box:
  - a. Enter a name for the system you want to register.

### i 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>-commerce-cloud`. This helps you identify the system type when assigning systems to a formation.

- b. In the *Type* dropdown list, select the system type.
- c. Choose *Register*.

## Results

Once you register a system, SAP Cloud Platform generates an integration token that you use to complete the integration on the respective SAP solution system side.

### ⚠ Caution

Kyma runtime has a default limit of 10 total connected systems and Namespaces. For example, you can reach this default limit by connecting 5 systems and creating 5 Namespaces, 8 systems and 2 Kyma Namespaces or any combination that equals 10. If you anticipate a total greater than 10, submit a support ticket or request a raise to the quota to 20. The support team will make the change and notify you when it is done. It typically takes about 1 day for the change to take effect.

## Next Steps

1. Copy the integration token and use it to configure the integration on the respective SAP system side.

- In the SAP Cloud Platform cockpit, assign the system to a formation on the [Formations](#) page, as follows:
  - 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 (SAP CX) system side. However, to enable the API access, you first need to complete the integration.
  - For systems of type *SAP S/4HANA Cloud* and *SAP SuccessFactors*, you first need to configure the integration on the respective SAP system side.

For more information, see: See: [Assigning SAP Systems to a Formation \[page 652\]](#).

## Related Information

[Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment \[page 654\]](#)

[Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment \[page 692\]](#)

[Extending SAP Customer Experience Products in the Kyma Environment \[page 705\]](#)

### 4.1.2 Assigning SAP Systems to a Formation

You can create a formation and assign to it the different SAP systems of the different SAP solutions that you want to extend in the context of the same business case.

#### Prerequisites

##### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

- You are a global account administrator.
- You have enabled the environment, Cloud Foundry and/or Kyma, for the subaccount you want to assign to a formation.
- You have registered the systems of the SAP solutions you want to assign to a formation. See [Registering an SAP System \[page 650\]](#).

#### Context

You can create a formation and assign to it the different SAP systems of the different SAP solutions that you want to extend.

Extension business cases often involve extending several SAP solutions at a time. For example, for a single business case you have to extend the functionality and/or the UI of:

- An SAP SuccessFactors system, and an SAP S/4HANA Cloud system. First, you need to configure the connectivity of each of these systems to SAP Cloud Platform for both Cloud Foundry and Kyma environments. Both extension applications 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 SAP Cloud Platform Kyma environment.
- A single system of the supported SAP solutions.

When creating a formation in the SAP Cloud Platform Cockpit, you assign to it the systems of the different SAP solutions you want to extend, and an existing subaccount. You do this configuration once but you can change it any time.

These are the system types that can take part of a single formation and the respective supported SAP Cloud Platform environment:

System Type	Cloud Foundry Environment	Kyma Environment
SAP S/4HANA Cloud	Supported	Supported
SAP SuccessFactors	Supported	Supported
SAP Commerce Cloud		Supported
SAP Cloud for Customer		Supported
SAP Field Service Management		Supported

## Procedure

1. Open SAP Cloud Platform Cockpit.
2. Navigate to your global account.
3. Choose  [System Landscape](#)  [Formations](#)  from the left hand-side navigation.
4. At the right top of the page, choose [Create Formation](#).
5. For the new formation you have to specify:
  - Unique name
  - A subaccount that you have previously created
  - All the systems of the different SAP systems that will be assigned to this formation
6. Choose [Create](#).

## 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, change the assigned systems, and the subaccount. The status of each system depends on whether you have registered that system in the SAP Cloud Platform global account. If you want to delete a formation, you need to unassign all systems and the subaccount in advance.

## Next Steps

For the systems of type SAP S/4HANA Cloud and SAP SuccessFactors and a formation using Cloud Foundry environment, you need to continue with assigning the required entitlements and creating a service instance to be able to consume the respective APIs.

- SAP S/4HANA Cloud:
  1. [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 661\]](#)
  2. [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 662\]](#)
- SAP SuccessFactors:
  1. [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 697\]](#)
  2. [Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API \[page 698\]](#)

### 4.1.3 Extending SAP S/4HANA Cloud in the Cloud Foundry and Kyma Environment

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

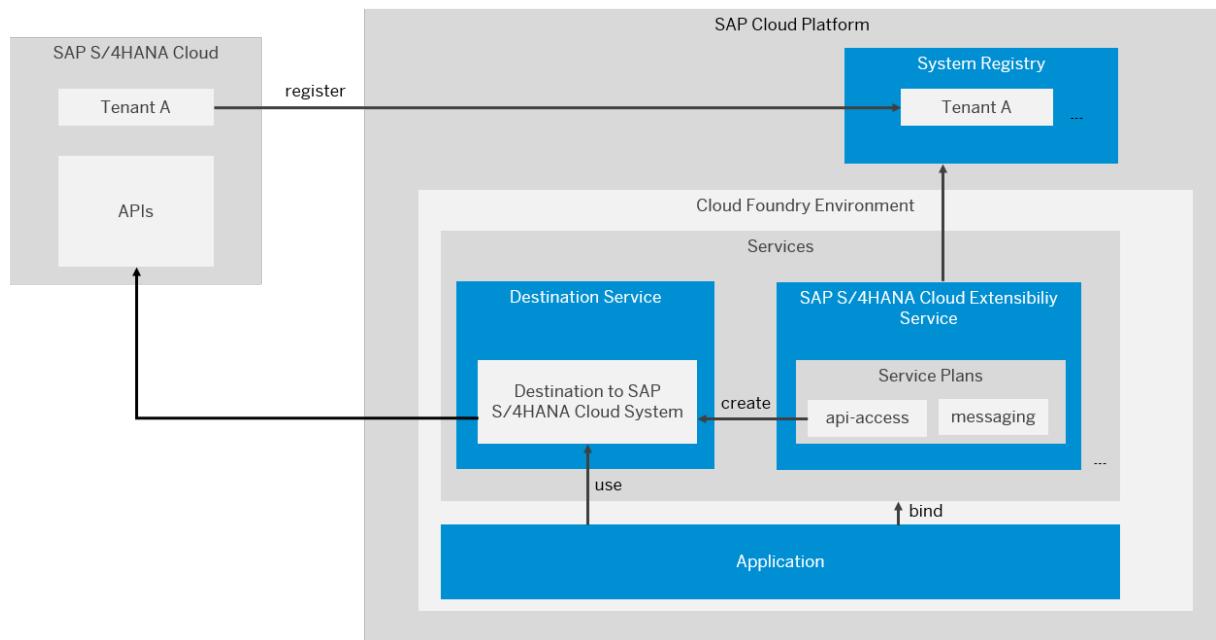
Extend SAP S/4HANA Cloud with extension applications running on the cloud platform using automated integration configuration.

## Introduction

SAP Cloud Platform offers 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.

The following graphic provides a high-level overview of the integration between SAP Cloud Platform Cloud Foundry environment and SAP S/4HANA Cloud:



## Process Flow

To integrate SAP Cloud Platform and SAP S/4HANA Cloud so that you can build extension applications, you need to:

Integrating SAP Cloud Platform and SAP S/4HANA Cloud

Process Step	Related Documentation
1. Connect the SAP S/4HANA Cloud system you want to extend with the corresponding SAP Cloud Platform global account.  During the pairing process you create an integration 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.	<a href="#">Register an SAP S/4HANA Cloud System in an SAP Cloud Platform Global Account [page 658]</a>

Process Step	Related Documentation
<p>2. Make the SAP S/4HANA Cloud system accessible in the SAP Cloud Platform subaccounts 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 Cloud Platform Subaccount [page 661]</a></p>

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 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 S/4HANA 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 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 Cloud Platform 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 662]</a></li> <li><a href="#">Enable the Consumption of SAP S/4HANA Cloud Events [page 679]</a></li> </ul>

## Related Information

[Cloud Management Tools - Feature Set Overview](#)

## 4.1.3.1 Register an SAP S/4HANA Cloud System in an SAP Cloud Platform Global Account

To connect an SAP S/4HANA Cloud system with an SAP Cloud Platform global account, you need to register the system in the corresponding SAP Cloud Platform global account.

### Prerequisites

- See [Registering an SAP System \[page 650\]](#)
- 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

The registration process is based on an integration token that is used for the pairing of the SAP S/4HANA Cloud system and the corresponding SAP Cloud Platform global account. You create the token in the SAP Cloud Platform 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.

The registration process has the following states displayed in the SAP Cloud Platform cockpit:

- *Pending* - the integration token for an SAP system has been created but the registration on the respective SAP S/4HANA Cloud system side has not been performed or completed.
- *Registered* - the integration token has been used and the automated registration process has been successfully completed.
- *Failed* - the registration has failed.

### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your global account, and then choose  [System Landscape](#) .
2. In the [Systems](#) panel, choose [Register System](#).
3. In the [Register System](#) dialog box:
  - a. Enter a name for the system you want to register.

#### Note

Use only printable ASCII characters.

- b. In the *Type* dropdown list, select the system type.
- c. Choose *Register*.

SAP Cloud Platform generates an integration token that the tenant administrator of the extended SAP S/4HANA Cloud system uses when configuring the integration between your SAP S/4HANA Cloud system and SAP Cloud Platform on the respective SAP S/4HANA Cloud system side.

4. Copy the integration 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 copy the integration token later, once the system appears in the list of registered systems.

#### Note

The integration 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. The integration token can be used only once, for registering a single SAP S/4HANA Cloud system.

5. Close the dialog box.

The SAP S/4HANA Cloud system appears in the list of registered systems. Its status is *Pending* because the registration process is not yet completed.

#### Note

While the system is still in status *Pending* you can delete the integration token and remove the system from the list of registered system. To do so, choose the  (*Unregister system*) button.

6. (Optional) For systems in status *Pending*, you can view and copy the integration token. To do so, choose the  (*Display token*) button.
7. 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 \[page 660\]](#).

#### Note

You can register a system only once with the same name 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 SAP Cloud Platform Extensions](#) in the SAP S/4HANA Cloud tenant. 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 SAP Cloud Platform cockpit for the corresponding system to be able to start the automated registration process again.

8. Check the status of the registration process. To do so, in the SAP Cloud Platform cockpit navigate to your global account, and in the *Systems* panel, check if the status of the SAP S/4HANA Cloud system has changed to *Registered*.

If you are already in the *Systems* panel, refresh the page to check if the status has changed.

## Next Steps

[Trigger the Registration in the SAP S/4HANA Cloud Tenant \[page 660\]](#)

### 4.1.3.1.1 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 SAP Cloud Platform global account.

#### 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 SAP Cloud Platform Extensions* tile.
2. In the *Maintain SAP Cloud Platform Extensions* screen, in the *Integrations* section, choose *New*.
3. In the *Integration Token* field, enter the content of the integration token from the SAP Cloud Platform 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*.
6. (Optional) If required, you can perform the following actions:
  - Disable the integration. To do so, choose the *Disable* button in the *Action* column.
  - Delete the integration. To do so, choose the *Delete* button in the *Actions* column.

#### i Note

Before deleting an integration, make sure you have deleted any SAP S/4HANA Cloud Extensibility service instances for this integration in the SAP Cloud Platform cockpit. See [Create a Service Instance to Consume the SAP S/4HANA Cloud APIs \[page 662\]](#).

## 4.1.3.2 Configure the Entitlements for the SAP Cloud Platform Subaccount

Configure the required entitlements to make the APIs of the SAP S/4HANA Cloud system which is registered in your SAP Cloud Platform global account accessible in the SAP Cloud Platform subaccount in which your extension applications will reside.

### Prerequisites

- You are an administrator of the global account.
- The subaccount is in the SAP Cloud Platform Cloud Foundry environment with enabled Cloud Foundry capabilities.

### Procedure

1. In SAP Cloud Platform cockpit, navigate to your global account.
2. In the navigation area, choose *Entitlements* *Subaccount Assignments*.
3. At the top of the page, select your subaccount from the drop down menu, choose *Go*, and then choose *Configure Entitlements*.

#### → Tip

If your global account contains more than 20 subaccounts, choose to open up the value help dialog. There you can filter subaccounts by role, environment and region to make your selection easier and faster. You can only select a maximum of 50 subaccounts at once.

4. Choose *Add Service Plans*, and then select the *SAP S/4HANA Cloud Extensibility* service.
5. In the *Available Service Plans* area, select the system you have registered and the required service plans, and then choose *Add Service Plan*.

#### i 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 \[page 691\]](#).

### Related Information

[Configuring the Entitlements for the SAP Cloud Platform Subaccount](#)

### 4.1.3.3 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 Cloud Platform 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 the SAP Cloud Platform, 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 Cloud Platform 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.

##### i 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 File - Properties \[page 667\]](#).

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 \[page 691\]](#)
- [Custom Communication Scenarios](#)

For more information about the service plans supported for the SAP S/4HANA Cloud integration, see [Supported Service Plans for SAP S/4HANA Cloud \[page 691\]](#).

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. SAP Cloud Platform supports the following authentication scenarios for the communication flow between SAP Cloud Platform 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 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](#).

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 \[page 663\]](#)
- [Create an SAP S/4HANA Cloud Extensibility Service Instance in the Kyma Environment \[page 666\]](#)

## Related Information

[Communication Arrangement JSON File - Properties \[page 667\]](#)

### 4.1.3.3.1 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 \[page 662\]](#).

## Context

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 S/4HANA Cloud system.

### i 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.

### i 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 Cloud Platform cockpit, navigate to the space 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 Cloud Platform, choose [SAP S/4HANA Cloud Extensibility](#).
5. In the navigation area, choose [Instances](#), and then choose [New Instance](#).
6. In the [Create Instance](#) wizard:
  - a. In the [Plan](#) dropdown list select the [api-access](#) service plan, and then in the [System Name](#) dropdown list select the SAP S/4HANA Cloud system that you have registered. Choose [Next](#).
  - b. To define the communication arrangement and the authentication type for the API access, specify a JSON file or specify parameters in the JSON format. Choose [Next](#).  
For more information about the structure of the JSON file, see [Communication Arrangement JSON File - Properties \[page 667\]](#).
  - c. (Optional) If you have already deployed an application that you want to bind to the new service instance, choose it from the list. Choose [Next](#).
  - d. Enter a name for your instance and choose [Finish](#).

After you have created the service instance:

- o The newly created instance appears in the list of instances in the [Instance](#) panel.
- o 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 Extensiblty service which is **s4-hana-cloud**.

For more information, see [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

#### i 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* panel.
- 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](#).

#### i 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 Cloud Platform 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.

#### i 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 \(Cloud Foundry Environment\)](#).

## 4.1.3.3.2 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 \[page 662\]](#).
- Configure the roles in the Kyma environment. See [Roles in the Kyma Environment \[page 1080\]](#)
- Have enabled the Kyma environment for the subaccount you are using. See [Provision Kyma Environment \[page 1083\]](#)
- Have configured the entitlements of the SAP S/4HANA Cloud Extensibility service. See [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 661\]](#)

### Context

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 S/4HANA Cloud system.

#### i 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 dashboard using the [Console URL](#).
3. In the Kyma dashboard, choose [Namespaces](#) from the left hand-side navigation and open the *default* Namespace.
4. Choose [Catalog](#) from the left hand-side navigation.
5. In the [Services](#) tab, search for the [SAP S/4HANA Cloud Extensibility](#) tile.
6. Open the [SAP S/4HANA Cloud Extensibility](#) tile and choose [Add](#) in the upper right-hand corner. A new dialog opens.
7. Fill in the fields of the SAP S/4HANA Cloud cluster service class:
  - Give a meaningful name of the new cluster service class.

- Select the [api-access](#) plan.
- Choose [Add parameters](#).
- To define the communication arrangement and the authentication type for the API access, specify a JSON file or specify parameters in the JSON format. For more information about the structure of the JSON file, see [Communication Arrangement JSON File - Properties \[page 667\]](#).
- Choose [Create](#).

## Next Steps

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 \[page 642\]](#).
- Consume the automatically generated destination. See [Consuming the Destination Service](#).

**i Note**

The name of the destination is the same as the name of the service instance you have created.

### 4.1.3.3.3 Communication Arrangement JSON File - Properties

Use the service JSON descriptor to define the communication arrangement and the authentication type for the SAP S4/HANA Cloud API access.

## Context

To construct the JSON file 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 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](#).

## Properties

Parameter	Description
systemName	<p>The name of the system you have registered in SAP Cloud Platform.</p> <p><b>i Note</b></p> <p>The system must be in status <i>Registered</i>.</p>
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>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>Max length: 80</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>i 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>i Note</b></p> <p>The generated X.509 certificate that is used to configure the trust between SAP Cloud Platform and SAP S/4HANA Cloud system expires two years after you have created the instance.</p>
outboundAuthentication	<p>A communicationArrangement property.</p> <p>The type of the authentication used by SAP S/4HANA Cloud to call SAP Cloud Platform APIs.</p> <p><b>i Note</b></p> <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> </ul>
	<p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes, if there is no inboundAuthentication defined</li> <li>• Allowed values: BasicAuthentication, OAuth2ClientCredentials,</li> </ul>

Parameter	Description
communicationSystem	<p>A communicationArrangement property.</p> <p>This represents the <i>Communication System</i> view of the communication arrangement.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
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 Cloud Platform, this is the URL of the extension application running on SAP Cloud Platform.</p> <p>This is equivalent to ► <i>Technical Data</i> ► <i>General</i> ► <i>Host Name</i> █ in the <i>Communication System</i> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes</li> </ul>
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 <b>443</b> port is used for the communication.</li> <li>• Type: String</li> <li>• Allowed values: positive integer [1-65535]</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 ► <i>Technical Data</i> ► <i>OAuth 2.0 Settings</i> ► <i>Auth. Endpoint</i> █ in the <i>Communication System</i> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>

Parameter	Description
<code>oAuthTokenEndpoint</code>	<p>A <code>communicationSystem</code> property.</p> <p>The OAuth token endpoint. This is equivalent to  <a href="#">Technical Data</a>  <a href="#">OAuth 2.0 Settings</a>  <a href="#">Token Endpoint</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>
<code>outboundCommunicationUser</code>	<p>A <code>communicationSystem</code> 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 data-bbox="794 864 1391 1021" style="background-color: #f0f8ff; padding: 10px;"> <p><b>i 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>
<code>username</code>	<p>A <code>outboundCommunicationUser</code> property.</p> <p>The username of the communication user.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes</li> </ul>
<code>password</code>	<p>A <code>outboundCommunicationUser</code> property.</p> <p>The password of the communication user.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: Yes</li> </ul>
<code>outboundServices</code>	<p>A <code>communicationArrangement</code> property.</p> <p>A list of outbound service objects.</p> <p>This is equivalent to the <a href="#">Outbound Services</a> section in the SAP S/4HANA Cloud <a href="#">Communication Arrangement</a> view.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>Required: No</li> </ul>

Parameter	Description
outboundService	<p>A <code>outboundServices</code> property.</p> <p>A specific outbound service object.</p> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: No</li> </ul>
name	<p>A <code>outboundService</code> 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</li> </ul>
urlPath	<p>A <code>outboundService</code> property.</p> <p>This is equivalent to the <a href="#">Path</a> 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 <code>outboundService</code> property.</p> <p>This is equivalent to the <a href="#">Service Status</a> 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>A <code>outboundService</code> property.</p> <p>A list of attribute objects.</p> <p>This is equivalent to the <a href="#">Additional Properties</a> 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 attributes property.</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 attribute property.</p> <p>The name of the additional property of the outbound service. It 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> <p><b>Rules/Guidelines</b></p> <ul style="list-style-type: none"> <li>• Required: Yes</li> </ul>
value	<p>An attribute 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</li> </ul>
attributes	<p>A communicationArrangement property.</p> <p>A list of attribute objects.</p> <p>This is equivalent to the <i>Additional Properties</i> 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 attributes property.</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> column in the properties 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</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</li> </ul>

## Related Information

[Communication Arrangement JSON File - Examples \[page 674\]](#)

### 4.1.3.3.4 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](#).

#### Example for Enabling Communication Scenario: Sales Order Integration (SAP\_COM\_0109)

This is an example of a JSON file for a communication arrangement with an inbound connection with *Basic Authentication*.

```
{
  "systemName": "DEMO",
  "communicationArrangement": {
    "communicationArrangementName": "INBOUND_COMMUNICATION_ARRANGEMENT",
```

```

        "scenarioId": "SAP_COM_0109",
        "inboundAuthentication": "BasicAuthentication"
    }
}

```

## Example for Enabling Communication Scenario: Product Integration (SAP\_COM\_0009)

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* with the optional property *port* specified for the outbound connection.

```

{
  "systemName": "DEMO",
  "communicationArrangement": {
    "communicationArrangementName": "INBOUND_COMMUNICATION_ARRANGEMENT",
    "scenarioId": "SAP_COM_0009",
    "inboundAuthentication": "BasicAuthentication",
    "outboundAuthentication": "BasicAuthentication",
    "outboundServices": [
      {
        "name": "Replicate Product from S/4 System to Client",
        "isServiceActive": false
      },
      {
        "name": "Product Master - Replicate from SAP S/4HANA to Client",
        "isServiceActive": false
      },
      {
        "name": "Product Master - Confirmation from SAP S/4HANA to Client",
        "isServiceActive": false
      }
    ],
    "communicationSystem": {
      "communicationSystemHostname": "default.com",
      "port": "80",
      "outboundCommunicationUser": {
        "username": "DefaultUser",
        "password": "DefaultPassword"
      }
    }
  }
}

```

## Example for Enabling Communication Scenario: Business Partner, Customer and Supplier Integration (SAP\_COM\_0008)

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": [
    {
        "name": "Replicate Customers from S/4 System to Client",
        "isServiceActive": false
    },
    {
        "name": "Replicate Suppliers from S/4 System to Client",
        "isServiceActive": false
    },
    {
        "name": "Replicate Company Addresses from S/4 System to Client",
        "isServiceActive": false
    },
    {
        "name": "Replicate Workplace Addresses from S/4 System to
Client",
        "isServiceActive": false
    },
    {
        "name": "Replicate Personal Addresses from S/4 System to Client",
        "isServiceActive": false
    },
    {
        "name": "Business Partner - Replicate from SAP S/4HANA Cloud to
Client",
        "isServiceActive": false
    },
    {
        "name": "Business Partner Relationship - Replicate from SAP S/
4HANA Cloud to Client",
        "isServiceActive": false
    },
    {
        "name": "Business Partner - Send Confirmation from SAP S/4HANA
Cloud to Client",
        "isServiceActive": false
    },
    {
        "name": "BP Relationship - Send Confirmation from SAP S/4HANA
Cloud to Client",
        "isServiceActive": false
    }
],
"communicationSystem": {
    "communicationSystemHostname": "default.com",
    "outboundCommunicationUser": {
        "username": "DefaultUser",
        "password": "DefaultPassword"
    }
}
}
}

```

## **Example for Enabling Communication Scenario: SAP Web IDE Integration (SAP\_COM\_0013)**

This is an example of a JSON file for a communication arrangement with an inbound connection with *OAuth2SAMLBearerAssertion* and an outbound connection with *NoAuthentication*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "communicationArrangementName": "0013_ARRANGEMENT",  
        "scenarioId": "SAP_COM_0013",  
        "inboundAuthentication": "OAuth2SAMLBearerAssertion",  
        "outboundAuthentication": "NoAuthentication",  
        "outboundServices": [  
            {  
                "name": "Launch SAP Web IDE",  
                "isServiceActive": false  
            }  
        ],  
        "communicationSystem": {  
            "communicationSystemHostname": "default.com"  
        }  
    }  
}
```

## **Example for Enabling Communication Scenario: SAP Cloud for Real Estate Contract Management Integration (SAP\_COM\_0213)**

This is an example of a JSON file for a communication arrangement with an inbound connection with authentication type *OAuth2SAMLBearerAssertion*.

```
{  
    "systemName": "DEMO",  
    "communicationArrangement": {  
        "communicationArrangementName": "0213_ARRANGEMENT",  
        "scenarioId": "SAP_COM_0213",  
        "inboundAuthentication": "OAuth2SAMLBearerAssertion"  
    }  
}
```

## **Example for Enabling Communication Scenario: SAP Watch List Screening - Screening Integration (SAP\_COM\_0219)**

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: Migration Cockpit Integration (SAP\_COM\_0259)

This is an example of a JSON file for a communication arrangement with an outbound connection with authentication type *BasicAuthentication*.

```

{
    "systemName": "DEMO",
    "communicationArrangement": {
        "outboundAuthentication": "BasicAuthentication",
        "communicationArrangementName": "0259_ARRANGEMENT",
        "scenarioId": "SAP_COM_0259",
        "outboundServices": [
            {
                "name": "Migration Cockpit: Connection to Staging Database",
                "isServiceActive": "true",
                "attributes": [
                    {
                        "name": "DATABASE CONNECTION NAME",
                        "value": "UniqueDBName"
                    },
                    {
                        "name": "SERVICE ID",
                        "value": "ServiceID"
                    }
                ]
            }
        ],
        "communicationSystem": {
            "communicationSystemHostname": "host",
            "outboundCommunicationUser": {
                "username": "DefaultUser",
                "password": "DefaultPassword"
            }
        }
    }
}

```

## 4.1.3.4 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 Cloud Platform Enterprise Messaging service.

### Prerequisites

- You have registered an SAP S/4HANA Cloud tenant in the SAP Cloud Platform global account. See [Register an SAP S/4HANA Cloud System in an SAP Cloud Platform Global Account \[page 658\]](#).
- You have configured the entitlements to make the registered SAP S/4HANA Cloud tenant accessible in the SAP Cloud Platform subaccount. See [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 661\]](#).

### Procedure

To configure the connectivity between SAP Cloud Platform Enterprise Messaging 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 Cloud Platform, 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 service plan to the SAP Cloud Platform subaccount you want to pair with the SAP S/4HANA Cloud tenant. See [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 661\]](#).

**i Note**

Currently, the *messaging* service plan is available only on SAP Cloud Platform, Cloud Foundry environment, Europe (Frankfurt) region.

2. Assign the required number of Enterprise Messaging instances to the SAP Cloud Platform subaccount you want to pair with the SAP S/4HANA Cloud tenant. See [Add Quota for the Enterprise Messaging to the SAP Cloud Platform Subaccount \[page 680\]](#).
2. Configure the connectivity between Enterprise Messaging and the SAP S/4HANA Cloud tenant. See [Set Up the Connectivity Between SAP Cloud Platform Enterprise Messaging and the SAP S/4HANA Cloud Tenant \[page 681\]](#).
3. Enable the Enterprise Messaging for your SAP Cloud Platform subaccount. See [Enable Enterprise Messaging Service for Your SAP Cloud Platform Subaccount \[page 688\]](#).

### Related Information

[SAP Cloud Platform Enterprise Messaging](#)

## 4.1.3.4.1 Add Quota for the Enterprise Messaging to the SAP Cloud Platform Subaccount

To be able to consume SAP S/4HANA Cloud events, you need to add quota for the SAP Cloud Platform Enterprise Messaging service to the SAP Cloud Platform subaccount.

### Context

The *messaging* service plan connects SAP S/4HANA Cloud tenant to the Enterprise Messaging service for the subaccount where the Enterprise Messaging entitlement is configured. This Enterprise Messaging service instance allows you to consume events from SAP S/4HANA Cloud.

You need a quota for the Enterprise Messaging service to create an Enterprise Messaging service instance. Then, you bind this service instance to an application to consume events from this application.

### Procedure

1. In SAP Cloud Platform 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*.

In the *Entitlements* panel, SAP Cloud Platform cockpit displays the list of all the resources you are entitled to use.

3. In the *Entitlements* panel, choose *Edit*.
4. Navigate to *Enterprise Messaging* service, and in the default service plan column, use the plus (+) and minus (-) buttons to adjust the quotas for the subaccount which you want to pair with the SAP S/4HANA Cloud tenant.

#### i Note

You can adjust quotas for only one subaccount at a time. To adjust quotas for multiple subaccounts, complete the changes for a single subaccount, save your changes and choose *Edit* again to continue with another subaccount.

5. Choose *Save*.

### Related Information

[Add Quotas to Subaccounts Using the Cockpit](#)

## 4.1.3.4.2 Set Up the Connectivity Between SAP Cloud Platform Enterprise Messaging and the SAP S/4HANA Cloud Tenant

Use this procedure to configure secure communication between SAP S/4HANA Cloud and SAP Cloud Platform Enterprise Messaging.

### Context

To configure the connectivity between the SAP S/4HANA Cloud tenant and Enterprise Messaging and to enable the exchange of credentials between the two systems, you first need to create an [SAP S/4HANA Cloud](#) service instance with service plan messaging. For more information about the messaging service plan, see [Supported Service Plans for SAP S/4HANA Cloud \[page 691\]](#).

When creating this service instance, you create the required configurations in both the SAP S/4HANA Cloud tenant and the Enterprise Messaging system associated with the SAP Cloud Platform subaccount, so that events can flow.

After you have created the [SAP S/4HANA Cloud](#) 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 Cloud Platform Enterprise Messaging service instance for the application to consume SAP S/4HANA Cloud events.

### Procedure

1. Create an SAP S/4HANA Cloud service instance of plan messaging.
  - a. In the SAP Cloud Platform cockpit, navigate to the space in which you want to create a service instance.
  - b. In the navigation area, choose ► [Services](#) ► [Service Marketplace](#) ▾.
  - c. In the [Service Marketplace](#) panel, choose [SAP S/4HANA Cloud](#).
  - d. In the navigation area, choose [Instances](#), and then choose [New Instance](#).
  - e. In the [Create Instance](#) wizard:
    1. In the [Plan](#) dropdown list select the [messaging](#) service plan, and then in the [System Name](#) dropdown list select the SAP system that you have paired. Choose [Next](#).
    2. 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. Choose [Next](#).  
For more information about the structure of the JSON file, see [SAP S/4HANA Cloud Service Descriptor JSON File \[page 682\]](#).
  3. Enter a name for your instance and choose [Finish](#).

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>"}}}'

```

The newly created instance appears in the list of instances in the *Instance* panel.

- Configure event topics for the channel inside SAP S/4HANA Cloud tenant. Use the channel you have specified in Enterprise Messaging service descriptor JSON file when configuring the parameters for the communication arrangement in SAP S/4/HANA Cloud tenant.

To configure the event topics, follow the steps described in [Configuration Steps within the SAP S/4HANA Cloud System](#) in the SAP S/4HANA Cloud documentation.

## Next Steps

[Enable Enterprise Messaging Service for Your SAP Cloud Platform Subaccount \[page 688\]](#)

### 4.1.3.4.2.1 SAP S/4HANA Cloud Service Descriptor JSON File

The SAP S/4HANA Cloud service descriptor defines details of a messaging client and needs to be provided when provisioning new *SAP S/4HANA Cloud* service instances with service plan *messaging*.

The *SAP S/4HANA Cloud* service descriptor is defined in a JSON structure. It contains 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 Cloud Platform Enterprise Messaging service.

Parameters required to activate the communication arrangement in SAP S/4/HANA Cloud tenant

Parameter	Description
systemName	The name of the system you have registered in the SAP Cloud Platform Extension Factory.
<b>Rules/Guidelines</b>	
	<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>

Parameter	Description
emClientId	<p>Using default patterns, it generates communication arrangement name, channel name, emname, namespace, when these parameters are not explicitly provided.</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	<p>Defines the communication arrangement for the SAP S4/HANA Cloud tenant.</p>
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>i 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 attributes 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>"Communication arrangement for integration with Enterprise Messaging for EM Client: &lt;emclientId&gt;"</code></li> </ul>
TOPIC SPACE	<p>An attributes 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 namespace in the SAP Cloud Platform Enterprise Messaging configuration must be the same as the topic space.</li> </ul>
MQTT_QOS	<p>An attributes 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 Cloud Platform Enterprise Messaging 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 Cloud Platform Enterprise Messaging at least once.</li> </ul> </li> <li>Default value: <code>1</code></li> </ul>

Parameter	Description
RECONNECT_ATTEMPTS	<p>An attributes 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 attributes 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 Cloud Platform Enterprise Messaging service

Parameter	Description
emname	<p>Specifies the name of the Enterprise Messaging client. It is used by SAP Cloud Platform Enterprise Messaging 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 messaging client.</p> <p><b>Rules/Guidelines:</b> The namespace in the SAP Cloud Platform Enterprise Messaging 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 messaging 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>

## Example

In this example, only the required properties are provided.

```
{  
    "systemName": "DEV",  
    "emClientId": "s4hc"  
}
```

This descriptor is equivalent to:

### i Note

All properties can be overriden 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}/#"  
                    ]  
                }  
            }  
        }  
    }  
}
```

## 4.1.3.4.3 Enable Enterprise Messaging Service for Your SAP Cloud Platform Subaccount

Use this procedure to enable the SAP Cloud Platform Enterprise Messaging service for the subaccount where your extension application will reside.

### Context

To enable the SAP Cloud Platform Enterprise Messaging service for your SAP Cloud Platform subaccount, you have to go through the following steps.

### Procedure

1. Prepare a JSON file that contains details of a messaging client. See [Enterprise Messaging Service Descriptor JSON File \[page 689\]](#).
2. Create an SAP Cloud Platform Enterprise Messaging service instance for the application to consume SAP S/4HANA Cloud events.
  - a. In the SAP Cloud Platform cockpit, navigate to the space in which you want to create a service instance.
  - b. In the navigation area, choose Services Service Marketplace New Instance, and then choose *Enterprise Messaging* in the *Service Marketplace* panel.
  - c. In the navigation area, choose *Instances*, and then choose *New Instance*.
  - d. In the *Create Instance* wizard:
    1. In the *Plan* dropdown list select the *default* service plan. Choose *Next*.
    2. Specify the JSON file you prepared in **Step 1**. Choose *Next*.
    3. (Optional) If you have already deployed an application that you want to bind to the new service instance, select it from the list. Choose *Next*.
    4. Enter a name for your instance and choose *Finish*.
3. Subscribe to the Enterprise Messaging Business Application. See [Subscribe to the Enterprise Messaging Business Applications](#).
4. Go to the Enterprise Messaging Business Application. See [step 3, Manage Queues](#).
5. Create a queue that is specific to your application. See [step 5, Manage Queues](#).
6. Subscribe this queue to the channel topic that SAP S/4HANA Cloud tenant uses to produce events. For example, `sap/S4HANAOD/TS02/BO/BusinessPartner/*`. See:
  - [Manage Queue Subscriptions](#)
  - [Maintain Event Topics for a Channel](#)
  - [Enterprise Messaging Service Descriptor JSON File \[page 689\]](#)

## 4.1.3.4.3.1 Enterprise Messaging Service Descriptor JSON File

The SAP Cloud Platform Enterprise Messaging service descriptor defines details of a messaging client and needs to be provided when provisioning new enterprise messaging service instances with service plan default.

Parameter	Description
emname	<p>Specifies the name of the Enterprise Messaging client. It is used by SAP Cloud Platform Enterprise Messaging to identify clients.</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: 100</li></ul> <p><b>Rules/Guidelines:</b> It is unique within a subaccount.</p>
namespace	<p>Namespace for the messaging client.</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>• Maximum length: 24</li><li>• It is unique within a subaccount</li><li>• Contains exactly three segments, for example <b>a/b/c</b>.</li></ul>
options	<p>Defines the access channels for the message client.</p>
management	<p>An options 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 options 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>

Parameter	Description
messaging	<p>An options 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>
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>
outboundFilter	<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>
<div style="background-color: #f0f0f0; padding: 10px;"> <p><b>i Note</b></p> <p>The namespace part of the value must be specified exactly as the namespace value of the SAP S/4HANA Cloud service descriptor JSON File described in <a href="#">SAP S/4HANA Cloud Service Descriptor JSON File [page 682]</a>.</p> </div>	

## Example

```
{
  "emname": "messaging",
```

```

    "namespace": "sap/scp/bpconsumer",
    "options": {
        "management": true,
        "messagingrest": true,
        "messaging": true
    },
    "rules": {
        "queueRules": {
            "outboundFilter": [
                "${namespace}/#"
            ]
        },
        "topicRules": {
            "outboundFilter": [
                "${namespace}/#",
                "sap/S4HANAOD/TS02/BO/BusinessPartner/#"
            ]
        }
    }
}

```

## Related Information

[Syntax for Service Descriptor in Enterprise Messaging documentation](#)

### 4.1.3.5 Supported Service Plans for SAP S/4HANA Cloud

The following service plans are available for SAP Cloud Platform subaccounts 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 API Business Hub](#).
- For more information about SAP S/4HANA Cloud communication scenarios see:
  - [Communication Management](#)
  - [Custom Communication Scenarios](#)

#### i 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 File - Properties \[page 667\]](#).

## 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.

### i Note

This service plan has been renamed from [sap\\_com\\_0092](#).

## 4.1.4 Extending SAP SuccessFactors in the Cloud Foundry and Kyma Environment

### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

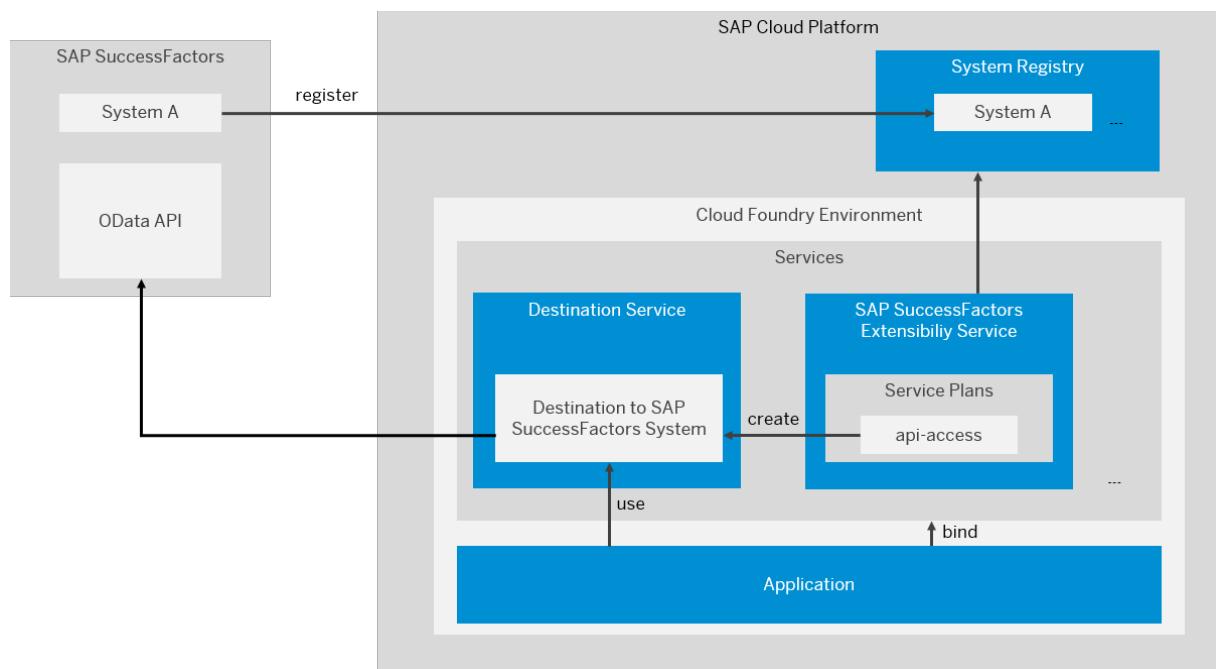
Use SAP Cloud Platform to extend SAP SuccessFactors with extension applications running on the cloud platform.

## Overview

SAP Cloud Platform 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.

The following graphic provides a high-level overview of the integration between SAP Cloud Platform Cloud Foundry environment and SAP SuccessFactors:



## Process Flow

To integrate SAP Cloud Platform and SAP SuccessFactors so that you can build extension applications, you need to perform the following tasks:

### Integrating SAP Cloud Platform and SAP SuccessFactors

Process Step	Related Documentation
1. Connect the SAP SuccessFactors system that you want to extend with the corresponding SAP Cloud Platform global account.	<a href="#">Register an SAP SuccessFactors System in an SAP Cloud Platform Global Account [page 695]</a> .
During the pairing process you create an integration token which is then used by the SAP SuccessFactors system tenant administrator to configure the integration on the SAP SuccessFactors system side.	

Process Step	Related Documentation
<p>2. Make the SAP SuccessFactors system accessible in the SAP Cloud Platform subaccounts in which you want to build your extension applications.</p> <p>To do so, you configure the entitlements and assign the corresponding quota where the extension applications will reside.</p>	<p><a href="#">Configure the Entitlements for the SAP Cloud Platform Subaccount [page 697]</a></p>
<p>3. Configure the communication flow.</p> <p>To do so, create a service instance of the SAP SuccessFactors Extensibility service using the <a href="#"><i>api-access</i></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 SuccessFactors system.</p> <p>SAP Cloud Platform supports the following authentication scenarios for SAP SuccessFactors:</p> <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>	<p><a href="#">Create a Service Instance to Consume the SAP SuccessFactors HXM Suite OData API [page 698]</a></p>
<p>4. Configure the Single-Sign On (SSO) between the SAP Cloud Platform subaccount 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 SAP Cloud Platform subaccount and the SAP SuccessFactors system using a SAML identity provider.</p>	<p><a href="#">Configure Single-Sign On Between SAP Cloud Platform Subaccount and SAP SuccessFactors [page 704]</a></p>

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

## 4.1.4.1 Register an SAP SuccessFactors System in an SAP Cloud Platform Global Account

To connect an SAP SuccessFactors system with an SAP Cloud Platform global account, you need to register the system in the corresponding SAP Cloud Platform global account.

### Prerequisites

- See [Registering an SAP System \[page 650\]](#).
- You are an administrator of the SAP Cloud Platform global account where you want to register your SAP SuccessFactors system.
- You have a dedicated SAP SuccessFactors company instance.
- To configure the integration on the SAP SuccessFactors system side, you need a user with permissions to access SAP SuccessFactors Provisioning.

### Context

The registration process is based on an integration token that is used for the pairing of the SAP SuccessFactors company and the corresponding SAP Cloud Platform global account. You create the token in the SAP Cloud Platform global account, and then start the automated registration process on the SAP SuccessFactors company side using this token.

The registration process has the following states displayed in the SAP Cloud Platform cockpit:

- *Pending* - the integration token for an SAP system has been created but the registration on the respective SAP system side has not been performed or completed.
- *Registered* - the integration token has been used and the automated registration process has been successfully completed.

### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your global account, and then choose ► [System Landscape](#) ➤ [Systems](#) ▾.
2. In the [Systems](#) panel, choose [Register System](#).
3. In the [Register System](#) dialog box:
  - a. Enter a name for the system you want to register.

#### i Note

Use only printable ASCII characters.

- b. In the [Type](#) dropdown list, select [SAP SuccessFactors](#).

- c. Choose *Register*.

SAP Cloud Platform generates an integration token that is used for triggering the automated integration on the SAP SuccessFactors company side. To use the token, you need a user with access to SAP SuccessFactors Provisioning.

4. Copy the integration token. The token is required for configuring the integration on the SAP SuccessFactors company side.

You can also copy the integration token later, once the system appears in the list of registered systems.

**i Note**

The integration 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 system. The integration token can be used only once, for registering a single SAP system.

5. Close the dialog box.

The SAP SuccessFactors system appears in the list of registered systems. Its status is *Pending* because the registration process is not yet completed.

**i Note**

While the system is still in status *Pending* you can delete the integration token and remove the system from the list of registered system. To do so, choose the  (*Unregister system*) button.

6. (Optional) For systems in status *Pending*, you can view and copy the integration token. To do so, choose the



(*Display token*) button.

7. Start the automated integration process on the SAP SuccessFactors company side:

If you do not have permissions to access SAP SuccessFactors Provisioning for the corresponding SAP SuccessFactors system, you need to send the integration token to a user with such permissions who will configure the integration on the SAP SuccessFactors system side.

- a. Open SAP SuccessFactors Provisioning.
- b. In the *List of Companies*, choose your SAP SuccessFactors company.
- c. In the *Edit Company Settings* section, choose *Extension Management Configuration*.
- d. In the *Integration Token* input field, paste the integration token.
- e. Choose *Add*.

Wait for the integration to finish. You can check the status of the process with the *Check Status* button next to your system name.

8. In the SAP Cloud Platform cockpit, check the status of the registration process. To do so, navigate to your global account, and in the *Systems* panel, check if the status of the SAP System has changed to *Registered*.

If you are already in the *Systems* panel, refresh the page to check if the status has changed.

**i Note**

You can register a system only once with the same name per global account.

## 4.1.4.2 Configure the Entitlements for the SAP Cloud Platform Subaccount

Configure the required entitlements to make the SAP SuccessFactors HXM Suite OData APIs of the SAP SuccessFactors system which is registered in your SAP Cloud Platform global account accessible in the SAP Cloud Platform subaccount in which your extension applications will reside.

### Prerequisites

- You are an administrator of the global account.
- The subaccount is in the SAP Cloud Platform Cloud Foundry environment with enabled Cloud Foundry capabilities.

### Procedure

1. In SAP Cloud Platform cockpit, navigate to your global account.
2. In the navigation area, choose  *Entitlements*  *Subaccount Assignments*.
3. Select your subaccount from the drop down menu, choose *Go*, and then choose *Configure Entitlements*.

#### → Tip

If your global account contains more than 20 subaccounts, choose  to open up the value help dialog. There you can filter subaccounts by role, environment and region to make your selection easier and faster. You can only select a maximum of 50 subaccounts at once.

4. Choose *Add Service Plans*, and then select the *SAP SuccessFactors Extensibility* service.
5. In the *Available Service Plans* area, select the system you have registered and the *api-access* service plan, and then choose *Add Service Plan..*

### Related Information

[Configuring the Entitlements for the SAP Cloud Platform Subaccount](#)

### **4.1.4.3 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 SAP Cloud Platform global account, you first need to create a service instance of the corresponding service.

#### **Context**

In the SAP Cloud Platform, for 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 Cloud Platform 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 Cloud Platform 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

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 \[page 699\]](#).
- For Kyma environment, see [Create an SAP SuccessFactors Extensibility Service Instance in the Kyma Environment \[page 701\]](#).

### 4.1.4.3.1 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 SAP Cloud Platform global account, you first need to create a service instance of the corresponding service.

#### Context

In the SAP Cloud Platform, 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 Cloud Platform 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 Cloud Platform 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.

#### i 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 Cloud Platform cockpit, navigate to the space 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 SAP Cloud Platform, choose [SAP SuccessFactors Extensibility](#).
5. In the navigation area, choose [Instances](#), and then choose [New Instance](#).
6. In the [Create Instance](#) wizard:

- a. In the *Plan* dropdown list select the *api-access* service plan, and then in the *System Name* dropdown list select the SAP SuccessFactors system that you have registered. Choose *Next*.
- b. To define the authentication type for the access to the SAP SuccessFactors HXM Suite API, specify a JSON file or specify parameters in the JSON format. Choose *Next*.  
For more information about the structure of the JSON file, see [Authentication Type JSON File \[page 703\]](#).
- c. (Optional) If you have already deployed an application that you want to bind to the new service instance, choose it from the list. Choose *Next*.
- d. Enter a name for your instance and choose *Finish*.

## Results

After you have created the service instance:

- The newly created instance appears in the list of instances in the *Instance* panel.
- 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](#).

### i 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](#) in the SAP Cloud Platform 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.

### i 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 \(Cloud Foundry Environment\)](#).

### 4.1.4.3.2 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 \[page 698\]](#).
- Configure the roles in the Kyma environment. See [Roles in the Kyma Environment \[page 1080\]](#)
- Have enabled the Kyma environment for the subaccount you are using. See [Provision Kyma Environment \[page 1083\]](#)
- Have configured the entitlements of the SAP SuccessFactors Extensibility service. See [Configure the Entitlements for the SAP Cloud Platform Subaccount \[page 661\]](#)

#### Context

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.

##### i 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 dashboard using the [Console URL](#).
3. In the Kyma dashboard, choose [Namespaces](#) from the left hand-side navigation and open the [default](#) namespace.

4. Choose *Catalog* from the left hand-side navigation.
5. In the *Services* tab, search for the *SAP SuccessFactors Extensibility* tile.
6. Open the *SAP SuccessFactors Extensibility* tile and choose *Add* in the upper right-hand corner. A new dialog opens.
7. Fill in the fields of the SAP SuccessFactors cluster service class:
  - Give a meaningful name of the new cluster service class.
  - Select the *api-access* plan.
  - Choose *Add parameters*.
  - To define the authentication type for the API access, specify a JSON file or specify parameters in the JSON format. For more information about the structure of the JSON file, see [Authentication Type JSON File \[page 703\]](#).
  - Choose *Create*.

## 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 \[page 642\]](#).
- Consume the automatically generated destination. See [Consuming the Destination Service](#).

**i Note**

The name of the destination is the same as the name of the service instance you have created.

### 4.1.4.3.3 Authentication Type 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 SAP Cloud Platform.</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: 100</li><li>• It is unique within a subaccount.</li><li>• The name must be the same as the one defined during the system registration in SAP Cloud Platform. See <a href="#">Register an SAP SuccessFactors System in an SAP Cloud Platform Global Account [page 695]</a>.</li></ul>
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>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",  
}
```

## 4.1.4.4 Configure Single-Sign On Between SAP Cloud Platform Subaccount and SAP SuccessFactors

Use this procedure to configure the Single-Sign On (SSO) between the SAP Cloud Platform subaccount 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 \[page 784\]](#).

### 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. 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 SAP Cloud Platform subaccount 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 SAP Cloud Platform account in the Cloud Foundry environment, you need to set up the single sign-on (SSO) according to this environment.

### Procedure

1. Establish Trust Between SAP SuccessFactors and SAP Cloud Platform. See [Establish Trust Between SAP SuccessFactors and SAP Cloud Platform \[page 705\]](#).
2. Register the Assertion Consumer Service of the SAP Cloud Platform Subaccount in SAP SuccessFactors. See [Register the Assertion Consumer Service of the SAP Cloud Platform Subaccount in SAP SuccessFactors \[page 727\]](#)

#### 4.1.4.4.1 Establish Trust Between SAP SuccessFactors and SAP Cloud Platform

Use this procedure to configure the SAP Cloud Platform subaccount trust settings 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>` 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 Cloud Platform cockpit.
  - a. Open the SAP Cloud Platform 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.

#### 4.1.5 Extending SAP Customer Experience Products in the Kyma Environment

You can configure the integration between SAP Cloud Platform and SAP Customer Experience (SAP CX) automatically to extend SAP CX products with applications running on the cloud platform.

##### Overview

###### Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

SAP Cloud Platform offers a standard way for extending SAP solutions.

You can extend SAP CX products in SAP Cloud Platform, Kyma environment 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 CX.

## Process Flow

To integrate SAP Cloud Platform and SAP CX products so that you can build extension applications, you need to perform the following tasks:

Integrating SAP Cloud Platform and SAP CX Products

Process Step	Related Documentation
1. Register an SAP CX system.	<a href="#">Register an SAP Customer Experience System [page 707]</a>
During the registration process you create an integration token which is then used in the SAP CX system for configuring the integration on the SAP CX side.	
2. Configure the integration on SAP CX side.  To do so, you pair the integration token with your SAP CX system.	<ul style="list-style-type: none"><li>For an SAP Commerce Cloud systems, see <a href="#">Step 2-5 in Retrieving Client Certificate</a></li><li>For an SAP Field Service Management system, see <a href="#">SAP Cloud Platform Extension Factory Integration</a>.</li><li>For an SAP Cloud for Customer system, see <a href="#">SAP Cloud Platform Extension Factory</a> ➤ <a href="#">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 CX system to a formation to enable the API access to the corresponding SAP C/HANA product's APIs.	<a href="#">Assigning SAP Systems to a Formation [page 652]</a>
4. Create a service instance of the required SAP CX Cloud services in the Kyma Service Catalog.	<a href="https://kyma-project.io/docs/components/service-catalog/#details-provisioning-and-binding">https://kyma-project.io/docs/components/service-catalog/#details-provisioning-and-binding</a>

### i Note

When the KymaRuntimeNamespaceAdmin binds a service to the Namespace, the credentials are made available to the KymaRuntimeNamespaceDevelopers who have access to this Namespace.

## Related Information

[Getting Started in the Kyma Environment \[page 50\]](#)

[Development in the Kyma Environment \[page 637\]](#)

[Administration and Operations in the Kyma Environment \[page 1079\]](#)

## 4.1.5.1 Register an SAP Customer Experience System

Register an SAP Customer Experience (SAP CX) system to connect it with an SAP Cloud Platform global account.

### Prerequisites

- See [Registering an SAP System \[page 650\]](#).
- You are an administrator of the global account where you want to register your SAP CX system.
- To configure the integration on the SAP CX system side, you need to be an administrator of the SAP CX tenant.

#### ⚠ Caution

Kyma runtime has a default limit of 10 total connected systems and Namespaces. For example, you can reach this default limit by connecting 5 systems and creating 5 Namespaces, 8 systems and 2 Kyma Namespaces, or any combination that equals 10. If you anticipate a total greater than 10, submit a support ticket or request a raise of the quota to 20. The support team will make the change and notify you when it is done. It typically takes about 1 day for the change to take effect.

### Context

The registration process is based on an integration token that is used for the pairing of the SAP CX system and the corresponding SAP Cloud Platform global account. You create the token in the SAP Cloud Platform cockpit, and then use it to configure the integration on the SAP CX system side.

The registration process has the following states displayed in the SAP Cloud Platform cockpit:

- *Pending* - the integration token for an SAP system has been created but the registration on the respective SAP CX system side has not been performed or completed. The system can be assigned to a formation on the [Formations](#) page in the SAP Cloud Platform cockpit but to enable the API access, you first need to configure the integration on the corresponding SAP CX product side.
- *Registered* - the integration 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 SAP Cloud Platform cockpit. You can use the corresponding SAP CX Cloud product APIs.
- *Failed* - the registration has failed.

### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your global account, and then choose  [System Landscape](#)  [Systems](#) 

2. In the *Systems* panel, choose *Register System*.
3. In the *Register System* dialog box:
  - a. Enter a name for the system you want to register.

**i Note**

Use only printable ASCII characters.

- b. In the *Type* dropdown list, select the system type.

The following SAP CX systems are supported:

- SAP Commerce Cloud
- SAP Field Service Management
- SAP Cloud for Customer

- c. Choose *Register*.

SAP Cloud Platform generates an integration token that you use to configure the integration between your SAP CX system and SAP Cloud Platform on the respective SAP CX system side.

4. Copy the integration token.

5. Close the dialog box.

The SAP CX system appears in the list of registered systems. Its status is *Pending* because the registration process is not yet completed.

6. (Optional) For systems in status *Pending*, you can generate a new token and copy it. To do so, choose the



(Display token) button.

7. Configure the integration on the SAP CX system side. See [Extending SAP Customer Experience Products in the Kyma Environment \[page 705\]](#).

**i Note**

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. If the token expires before you have

used it, you can generate a new token by choosing the (Display token) button. The integration token can be used only once, for registering a single SAP CX system.

8. Check the status of the registration process. To do so, in the SAP Cloud Platform cockpit navigate to your global account, and in the *Systems* panel, check if the status of the SAP CX 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 CX system, all of the exposed services and events are propagated to the Kyma runtime.

## 4.2 Extending SAP Solutions Using Manual Configurations

If your scenario requires a configuration setup that is not included in the automated extension configurations, you can optionally configure the integration between SAP Cloud Platform and your SAP solution manually.

### i Note

Use the manual configuration process only if your scenario is not included in the automated configurations with SAP Cloud Platform. See [Extending SAP Solutions Using Automated Configurations \[Feature Set A\] \[page 648\]](#)

If you configure the integration between SAP Cloud Platform and your SAP solution manually, you will not be able to benefit from any of the automations provided in the automated configurations.

### Process Flow

When configuring the integration between SAP Cloud Platform and an SAP solution manually, you need to perform the following tasks:

1. Configure the extension subaccount. To do so, you configure the trust between the extended SAP solution and SAP Cloud Platform.
2. Install and configure the extension application.

### Supported SAP Solutions

Using manual configurations, you can extend the following SAP solutions on SAP Cloud Platform, Cloud Foundry environment:

- SAP S/4HANA Cloud. See [Extending SAP S/4HANA Cloud in the Cloud Foundry Environment Manually \[page 709\]](#).
- SAP SuccessFactors. See [Extending SAP SuccessFactors in the Cloud Foundry Environment Manually \[page 725\]](#).
- SAP Cloud for Customer. See [Extending SAP Cloud for Customer in the Cloud Foundry Environment Manually \[page 733\]](#).

### 4.2.1 Extending SAP S/4HANA Cloud in the Cloud Foundry Environment Manually

This section guides you through the configuration tasks that you need to perform to enable the SAP Cloud Platform, Cloud Foundry environment for developing extension applications for your SAP S/4HANA Cloud tenant.

## Process Flow

### i Note

Use the manual configuration steps only if your scenario is not covered by the automated process with SAP Cloud Platform Extension Factory. If you configure the integration between SAP Cloud Platform and SAP S/4HANA Cloud manually, you may not be able to benefit from any automatons provided by SAP Cloud Platform Extension Factory.

You have to configure the cloud platform extension integration with SAP S/4HANA Cloud to enable the use of applications running on top of the platform from SAP S/4HANA Cloud.

These are the steps you need to follow:

1. Set up the SAP Cloud Platform extension subaccount by configuring the single sign-on. See [SAP Cloud Platform Extension Onboarding for SAP S/4HANA Cloud \[page 710\]](#).
2. Configure the extension application and create the respective destination in the SAP Cloud Platform cockpit. See [Extension Application Configuration \[page 713\]](#).

### 4.2.1.1 SAP Cloud Platform Extension Onboarding for SAP S/4HANA Cloud

The SAP Cloud Platform Cloud Foundry environment extension onboarding enables the integration between the cloud platform and SAP S/4HANA Cloud. This allows the SAP S/4HANA Cloud side-by-side extension applications to run on top of the cloud platform.

SAP Cloud Platform Identity Authentication service provides authentication, single sign-on, and on-premise integration. Identity Authentication service is closely integrated with SAP Cloud Platform, and it is offered as part of the platform.

The authentication to SAP S/4HANA Cloud applications is restricted to the authorized users. The identification of a user is verified by the identity provider, as specified by SAML 2.0. Identity Authentication service stores a list of all users that are allowed to access the service provider (SAP S/4HANA Cloud) along with their credentials. The integration between the SAP S/4HANA Cloud and the Identity Authentication service is based on a trust configuration. When a user attempts to access SAP S/4HANA Cloud 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 S/4HANA Cloud 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 SAP Cloud Platform subaccount and the SAP S/4HANA Cloud tenant using a SAML identity provider, for example SAP Cloud Platform Identity Authentication service. The single sign-on requires both solutions to be configured as trusted SAML service providers for the Identity Authentication service, and at the same time, the Identity Authentication service to be configured as trusted identity provider for the two solutions.

### i Note

You own an SAP S/4HANA Cloud tenant with a SAP Cloud Platform Identity Authentication tenant configured. You need to use the same Identity Authentication tenant for your SAP Cloud Platform subaccount.

## 4.2.1.1.1 Configure Single Sign-On with the Identity Authentication Service

To use the SAML 2.0 bearer assertion authentication for the communication flow between the extension application and SAP S/4HANA Cloud, you need to configure single-sign on (SSO).

If you have an SAP Cloud Platform account in the Cloud Foundry environment, you need to set up the single sign-on (SSO) according to this environment. The single sign-on requires both solutions, SAP Cloud Platform and SAP S/4HANA Cloud, to be configured as trusted SAML service providers for the Identity Authentication service, and at the same time, the Identity Authentication service to be configured as trusted identity provider for the two solutions. All the necessary configuration steps are described in [Establish Trust Between Identity Authentication Service and SAP Cloud Platform \[page 711\]](#).

### 4.2.1.1.1.1 Establish Trust Between Identity Authentication Service and SAP Cloud Platform

Use this procedure to configure the SAP Cloud Platform trust settings and add the SAP Cloud Platform Identity Authentication tenant as an identity provider.

#### Context

Identity Authentication service is closely integrated with SAP Cloud Platform, and it is offered as part of most of the cloud platform packages. For those packages the trust between the SAP Cloud Platform subaccount and Identity Authentication service is configured automatically and the tenant for Identity Authentication service is set up by default, once you have a partner or customer subaccount. However, you can manually configure the trust and set up the Identity Authentication tenant if your scenario requires it.

#### Procedure

1. In the Identity Authentication tenant, access the administration console by using the console's URL. You need to use another browser, or incognito session of the same browser.

The URL has the `https://<tenant ID>.accounts.onDemand.com/admin` pattern.

You can also get the URL from the Identity Authentication tenant registration e-mail.

2. You have to save the metadata of your Identity Authentication tenant on your local file system as an XML file. You can either find the tenant at [https://<tenant\\_ID>.accounts.ondemand.com/saml2/metadata](https://<tenant_ID>.accounts.ondemand.com/saml2/metadata) or access it via [Applications & Resources](#) [Tenant Settings](#) [SAML 2.0 Configuration](#). Then choose the [Download Metadata File](#) link. You will need this metadata in **Step 5**.
3. Open the SAP Cloud Platform cockpit and select the region in which your subaccount is hosted. Select the global account that contains your subaccount, and then choose the tile of your subaccount. For more information about regions, see [Regions and Hosts](#).
4. Choose [Security](#) [Trust Configuration](#).
5. Choose [New Trust Configuration](#) and fill in the required fields:
  - a. In the *Metadata* field, upload the metadata file you have downloaded from the Identity Authentication tenant in **step 2**. All the required fields are automatically filled in.
  - b. In the *Name* field, enter a meaningful name of your choice.
  - c. Choose [Save](#).
6. In the SAP Cloud Platform cockpit, download the service provider SAML metadata file. Open the link [https://<subaccount\\_subdomain>.authentication.<region\\_host>/saml/metadata](https://<subaccount_subdomain>.authentication.<region_host>/saml/metadata), where:
  - <subaccount\_subdomain> is part of the SAP Cloud Platform subaccount details in the cockpit.
  - <region\_host> is the API endpoint without api.cf.. See [Regions and Hosts](#).

When you are prompted, save the XML file on your local file system. This file contains the SAML 2.0 metadata describing SAP Cloud Platform as a service provider.

You will need this metadata in **Step 11**.

7. In the Identity Authentication tenant, access the administration console by using the console's URL. You need to use another browser, or incognito session of the same browser.

The URL has the [https://<tenant\\_ID>.accounts.ondemand.com/admin](https://<tenant_ID>.accounts.ondemand.com/admin) pattern.

You can also get the URL from the Identity Authentication tenant registration e-mail.

8. Choose [Applications & Resources](#) [Applications](#).
9. Choose the [+Add](#) button on the left-hand panel, and enter the name of your SAP Cloud Platform subaccount.
10. Choose [Save](#).
11. Configure the SAML 2.0 trust with SAP Cloud Platform subaccount as a service provider. To do so, proceed as follows:
  - a. On the left-hand side, choose the newly created application, and then choose [Trust](#).
  - b. Choose [SAML 2.0 Configuration](#).
  - c. Upload the metadata XML file of your SAP Cloud Platform subaccount that you have downloaded in **Step 6**.

On service provider metadata upload, the fields are populated with the parsed data from the XML file.

- d. Save the configuration settings.

## Results

The trust will be established automatically upon registration on both the SAP Cloud Platform subaccount and Identity Authentication tenant side.

## Related Information

[Get Started with Identity Authentication](#)  
[ID Federation with a Identity Authentication Tenant](#)

### 4.2.1.2 Extension Application Configuration

Use this procedure to configure the authentication flow between the extension application and the SAP S/4HANA Cloud system.

When configuring the SAP Cloud Platform extension application and you have SAP Cloud Platform subaccount in the Cloud Foundry environment, you have to set up the connectivity from the SAP Cloud Platform subaccount to the SAP S/4HANA Cloud tenant. To do that, you can use different authentication methods: Basic, Client Certificate, or SAML Bearer Assertion.

## Related Information

[Configure Application Authentication \[page 713\]](#)  
[Configuring the Extension Application's Connectivity to SAP S/4HANA Cloud \[page 714\]](#)

### 4.2.1.2.1 Configure Application Authentication

Use this procedure to configure the application with which you want to extend your SAP S/4HANA Cloud system.

## Prerequisites

- Have the Cloud Foundry command line interface (cf CLI) downloaded and installed. See [Download and Install the Cloud Foundry Command Line Interface](#).
- Have Node.JS including its NPM Packager Manager installed. See <https://nodejs.org/en/> ↗.

## Context

You need your own application router that connects your extension application to the centrally provided user account and authentication (UAA) service. This means that you need to deploy an approuter as part of your application that manages the user authentication for you.

The approuter has these main functions:

- Handles authentication for all apps of the application
- Serves static resources
- Performs route mapping (URL mapping)
- In case of multitenancy, it derives the tenant information from the URL and provides it to the extended services for User Account and Authentication service (XSUAA) to redirect the authentication request to the tenant-specific identity provider. See [User Account and Authentication Service of the Cloud Foundry Environment](#).

## Procedure

1. To set up the application router as part of your application, execute the following command in cf CLI:

```
npm config set @sap:registry=https://npm.sap.com
```

2. Create an xsuaa service instance. See [Create a Service Instance from the xsuaa Service](#).
3. Add an application router. See [Application Router and Destination](#)
4. Update the extension application by using the cf push command in the cf CLI.
5. Verify that there is an authentication for your application and access the application using the approuter URL.
  - a. Get the url of the approuter using the cf apps command.
  - b. Enter the approuter URL in a browser.
  - c. You are redirected to the Identity Authentication service that you have already configured. See [Establish Trust Between Identity Authentication Service and SAP Cloud Platform \[page 711\]](#).
  - d. After a successful login, you will get redirected to the welcome page of your extension application if you have defined one.

### 4.2.1.2.2 Configuring the Extension Application's Connectivity to SAP S/4HANA Cloud

Use this procedure to set up the connectivity between the SAP Cloud Platform application and the SAP S/4HANA Cloud system.

To set up the connectivity from an SAP Cloud Platform subaccount to an SAP S/4HANA Cloud tenant, you need to create HTTP destinations in the SAP Cloud Platform cockpit. These destinations provide data communication via HTTP protocol.

When creating an HTTP destination, you can use different authentication types for access control:

## Basic Authentication

Using authentication method you need to provide username and password. When configuring the HTTP destination in the cockpit, the username and password must correspond to the communication user in the SAP S/4HANA Cloud tenant.

See [Using Basic Authentication \[page 716\]](#).

## Client Certificate Authentication

### i Note

Available only for the manual configuration.

To configure a client certificate authentication, you need a client certificate signed by a trusted certificate authority (CA). You upload the public key when creating a communication user in the SAP S/4HANA Cloud tenant, and then, you add the corresponding keystore to the HTTP destination in the SAP Cloud Platform cockpit.

See [Using Client Certificate Authentication \[page 718\]](#).

## SAML Bearer Assertion Authentication

You can use the SAML Bearer assertion flow for consuming OAuth-protected resources. Users are authenticated by using SAML against the configured trusted identity providers. The SAML assertion is then used to request an access token from an OAuth authorization server. This access token must be added as an Authorization header with value `Bearer <access token>` in all HTTP requests to the OAuth-protected resources.

See [Using SAML Bearer Assertion Authentication \[page 720\]](#).

## Related Information

[Authentication for Java Resource Servers](#)

[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

## 4.2.1.2.2.1 Using Basic Authentication

### Context

To be able to use Basic authentication, you need to configure both SAP S/4HANA Cloud and SAP Cloud Platform sides.

### 4.2.1.2.2.1.1 Set Up SAP S/4HANA Cloud Side

### Context

From the SAP S/4HANA Cloud side you need to maintain the communication settings to configure the connectivity between SAP S/4HANA Cloud and SAP Cloud Platform.

### Procedure

1. Create a communication user. See [Maintain Communication Users](#).

For the basic authentication, you need to provide a password when defining the required credentials.

2. Create a communication system. See [Maintain Communication Systems](#).

- a. Log into the SAP Fiori launchpad in the SAP S/4HANA Cloud system.
- b. Select the *Communication Systems* tile.
- c. Choose *New* to create a new system.
- d. Enter a system ID and a system name.
- e. Choose *Create*.
- f. Enter information about SAP Cloud Platform in the *Technical Data* section.
- g. Choose *+* (Add) under *User for Inbound Communication*.
- h. In the dialog box that appears, select *User Name and Password* from the *Authentication Method* drop-down list.

The username corresponds to the communication user.

- i. Choose *OK* to confirm.
- j. Choose *Save*.

3. Create a communication arrangement and select a communication scenario. See [Maintain Communication Arrangements](#).

## Related Information

Communication Management

### 4.2.1.2.2.1.2 Set Up SAP Cloud Platform Side

#### Prerequisites

You have logged into the SAP Cloud Platform cockpit from the SAP Cloud Platform landing page for your subaccount.

#### Procedure

1. In the cockpit, go to the *Subaccounts* drop-down menu and choose your subaccount.
2. Choose *Connectivity* *Destinations* in the navigation panel.
3. Choose *New Destination* and fill in the following properties:

Parameter	Value
Name	Enter a meaningful name.
Type	<b>HTTP</b>
Description	(Optional) Enter a meaningful description.
URL	The service URL from the communication arrangement.
Proxy Type	<b>Internet</b>
Authentication	<b>Basic Authentication</b>
User	The name of the communication user you have in the SAP S/4HANA Cloud tenant.
Password	The password for the communication user.

4. Select the *Use default JDK truststore* checkbox.
5. Save your entries.

## Related Information

[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

### 4.2.1.2.2.2 Using Client Certificate Authentication

#### Context

To be able to use client certificate authentication, you need to configure both SAP S/4HANA Cloud and SAP Cloud Platform sides.

#### 4.2.1.2.2.2.1 Set Up SAP S/4HANA Cloud Side

#### Context

To use client certificate authentication, you first start with creating and configuring the communication settings in the SAP S/4HANA Cloud tenant. To do that, you have to:

#### Procedure

1. Obtain a client certificate signed by a trusted certificate authority (CA) in [.pem](#) format.  
You can find a list of the trusted CA in the SAP S/4HANA Cloud tenant using the **Maintain Certificate Trust List** application. See [Maintain Certificate Trust List](#).
2. Create a communication user and upload the public key. See [Maintain Communication Users](#).
3. Create a communication system and add the communication user as *User for Inbound Communication* with an authentication method SSL Client Certificate.
  - a. Log into the SAP Fiori launchpad in the SAP S/4HANA Cloud system.
  - b. Select the *Communication Systems* tile.
  - c. Choose *New* to create a new system.
  - d. Enter a system ID and a system name.
  - e. Choose *Create*.
  - f. Enter information about SAP Cloud Platform in the *Technical Data* section.
  - g. Choose *+* (Add) under *User for Inbound Communication*.

- h. In the dialog box that appears, select *SSL Client Certificate* from the *Authentication Method* drop-down list.
  - The username corresponds to the communication user.
  - i. Choose *OK* to confirm.
  - j. Choose *Save*.
4. Create a communication arrangement using an existing or create a new scenario that supports client certificate authentication. See [Maintain Communication Arrangements](#).

You can use the already created communication system. The settings in the *Inbound Communication* section are filled in automatically. Save the value from the *URL* field, you will need it when creating a destination in the SAP Cloud Platform subaccount.

## 4.2.1.2.2.2 Set Up SAP Cloud Platform Side

### Prerequisites

You have logged into the SAP Cloud Platform cockpit from the SAP Cloud Platform landing page for your subaccount.

### Procedure

1. In the cockpit, navigate to your *Subaccounts*.
2. Choose *Connectivity* *Destinations* in the navigation panel.
3. Choose *New Destination* and fill in the following properties:

Parameter	Value
Name	Enter a meaningful name.
Type	<b>HTTP</b>
Description	(Optional) Enter a meaningful description.
URL	The service URL from the communication arrangement.  Make sure you use the HTTPS protocol, otherwise the <i>ClientCertificateAuthentication</i> option would not appear in the <i>Authentication</i> dropdown list.
Proxy Type	<b>Internet</b>

Parameter	Value
Authentication	<b>ClientCertificateAuthentication</b>

**i Note**

Once you have selected this option, the system displays the [Upload and Delete Certificate](#) link.

4. Choose [Upload and Delete Certificate](#) link to upload your keystore. The keystore format .jks. When you finish uploading, choose [Close](#).

The keystore contains the key/pair signed by the trusted certificate authority (CA) in [Set Up SAP S/4HANA Cloud Side \[page 718\]](#), **step 1**.

- a. From the [Key Store Location](#) drop-down menu, select your keystore.
- b. In the [Key Store Password](#), enter the keystore password.
5. Select the [Use default JDK truststore](#) checkbox.
6. Save your entries.

## Related Information

[Managing Destinations](#)

### 4.2.1.2.2.3 Using SAML Bearer Assertion Authentication

#### Prerequisites

You have configured single-sign on (SSO) Identity Authentication Service. See [Configure Single Sign-On with the Identity Authentication Service \[page 711\]](#).

#### Context

To be able to use SAML Bearer Assertion authentication, you need to configure both SAP S/4HANA Cloud and SAP Cloud Platform sides.

## Related Information

[SAML Bearer Assertion Authentication](#)

### 4.2.1.2.2.3.1 Upload Your Key-Pair Keystore in SAP Cloud Platform

To use SAML Bearer Assertion authentication for the communication between your extension application and the SAP S/4HANA Cloud system, you first need to upload your key-pair keystore in SAP Cloud Platform on subaccount level.

#### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your extension subaccount in the Cloud Foundry environment.
2. Choose .
3. Choose .
4. Browse to the keystore file.

#### i Note

You can only use JKS, PFX and P12 files. The keystore file must contain exactly one key pair entry.

The keystore file containing your key pair entry is added.

## Related Information

[Use Destination Certificates](#)

### 4.2.1.2.2.3.2 Set Up SAP S/4HANA Cloud Side

#### Context

From the SAP S/4HANA Cloud side you need to maintain the communication settings to configure the connectivity between SAP S/4HANA Cloud and SAP Cloud Platform.

### i Note

Make sure you have assigned the appropriate business catalogs to the propagated business users in the SAP S/4HANA Cloud tenant.

## Procedure

1. Create a communication user. See [Maintain Communication Users](#).

For the SAML Bearer Assertion authentication, you need to provide a password when defining the required credentials.

2. Create a communication system. See [Maintain Communication Systems](#).

- a. Log into the SAP Fiori launchpad in the SAP S/4HANA Cloud system.
- b. Select the *Communication Systems* tile.
- c. Choose *New* to create a new system.
- d. Enter a system ID and a system name.
- e. Choose *Create*.
- f. In the *Technical Data* section, in the *Host Name* field, enter the URL of the Cloud Foundry application or service required by your scenario or consult your scenario configuration guide.

For example, the URL for an application deployed on SAP Cloud Platform in the Cloud Foundry environment is `https://<application_route>.cfapps.<location>.hana.ondemand.com`.

Where:

- o The domain depends on your location, in the European region for example, the domain is `cfapps.eu10.hana.ondemand.com`
  - o The application route is the single point of entry of an application. To view the application route, you can use the `cf apps` or `cf routes` of CLI command.
- g. Enable the OAuth Identity Provider by checking the box under *OAuth 2.0 Identity Provider*.
  - h. Upload the public key contained in the keystore file you have uploaded in your SAP Cloud Platform subaccount. See [Upload Your Key-Pair Keystore in SAP Cloud Platform \[page 721\]](#).
  - i. In the *Provider Name* field, enter a unique provider name. For example, you can enter the value of the CN field of your public key.
  - j. Choose *+* (Add) under *User for Inbound Communication*.
  - k. In the dialog box that appears, select *User Name and Password* from the *Authentication Method* drop-down list.

The username corresponds to the communication user.

- l. Choose *OK* to confirm.
- m. Choose *Save*.

3. Create a communication arrangement and select a communication scenario. See [Maintain Communication Arrangements](#).

### i Note

When you have the communication arrangement created, choose *OAuth 2.0 Details..*. Copy and save locally the fields and their values. You will need them when setting up the destination in the SAP Cloud Platform cockpit.

### 4.2.1.2.2.3.3 Set Up SAP Cloud Platform Side

#### Prerequisites

You have logged into the SAP Cloud Platform cockpit from the SAP Cloud Platform landing page for your subaccount.

#### Procedure

1. In the cockpit, go to the *Subaccounts* drop-down menu and choose your subaccount.
2. Choose *Connectivity* *Destinations* in the navigation panel.
3. Create an HTTP destination.

To enable principal propagation, create an *OAuth2SAMLBearerAssertion* HTTP destination and configure its settings as follows:

1. Configure the basic settings:

Parameter	Value
Name	Enter a meaningful name.
Type	<b>HTTP</b>
Description	(Optional) Enter a meaningful description.
URL	The service URL from the communication arrangement.
<b>i Note</b>	
Make sure you use the HTTPS protocol.	
Proxy Type	<b>Internet</b>
Authentication	<b>OAuth2SAMLBearerAssertion</b>
Key Store Location	In the dropdown list, select the key-pair keystore file you have uploaded in <a href="#">Upload Your Key-Pair Keystore in SAP Cloud Platform [page 721]</a> .

Parameter	Value
Key Store Password	The password for the keystore. <b>i Note</b> The password for the keystore must be the same as the one for the key pair entry in the keystore file.
Audience	The URL of your SAP S/4HANA Cloud account. To get it, log on to your SAP S/4HANA Cloud account. Select the profile picture and then choose <i>Settings</i> and copy the value from the <i>Server</i> field. <b>i Note</b> Make sure you use the HTTPS protocol.
Client Key	The name of the communication user you have in the SAP S/4HANA Cloud tenant.
Token Service URL	This is the <i>Token Service URL</i> from the <i>OAuth 2.0 Details</i> in the communication arrangement. See <a href="#">Set Up SAP S/4HANA Cloud Side [page 721]</a> , step 3.
Token Service User	The name of the communication user you have in the SAP S/4HANA Cloud tenant.
Token Service Password	The password for the communication user.
System User	This parameter is not used, leave the field empty.

2. Configure the required additional property. To do so, in the *Additional Properties* panel, choose *New Property*, and enter the following property:

Parameter	Value
authnContextClassRef	<b>urn:oasis:names:tc:SAML:2.0:ac:classes:X509</b>
assertionIssuer	Issuer of the SAML assertion. Enter the provider name you have specified when creating the communication system in SAP S/4HANA Cloud. See <a href="#">Set Up SAP S/4HANA Cloud Side [page 721]</a> .

3. Select the *Use default JDK truststore* checkbox.  
4. Save your entries.

## Related Information

[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

### 4.2.2 Extending SAP SuccessFactors in the Cloud Foundry Environment Manually

This section guides you through the configuration tasks that you need to perform to enable the SAP Cloud Platform, Cloud Foundry environment for developing extension applications for your SAP SuccessFactors system.

#### Process Flow

You have to configure the cloud platform extension integration with SAP SuccessFactors to enable the use of applications running on top of the platform from SAP SuccessFactors.

These are the steps you need to follow:

1. [Configuring the SAP Cloud Platform Subaccount for SAP SuccessFactors \[page 725\]](#)
2. [Installing and Configuring SAP SuccessFactors Extension Applications for Cloud Foundry Environment \[page 728\]](#)

#### 4.2.2.1 Configuring the SAP Cloud Platform Subaccount for SAP SuccessFactors

Use this procedure to configure the Single-Sign On (SSO) between AP Cloud Platform subaccount and the SAP SuccessFactors system.

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. 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 SAP Cloud Platform subaccount 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 SAP Cloud Platform account in the Cloud Foundry environment, you need to set up the single sign-on (SSO) according to this environment.

## Related Information

[Establish Trust Between SAP SuccessFactors and SAP Cloud Platform \[page 705\]](#)

[Register the Assertion Consumer Service of the SAP Cloud Platform Subaccount in SAP SuccessFactors \[page 727\]](#)

### 4.2.2.1.1 Establish Trust Between SAP SuccessFactors and SAP Cloud Platform

Use this procedure to configure the SAP Cloud Platform subaccount trust settings 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>` 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 Cloud Platform cockpit.
  - a. Open the SAP Cloud Platform cockpit and navigate to your subaccount.
  - b. Choose  .
  - 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.

## 4.2.2.1.2 Register the Assertion Consumer Service of the SAP Cloud Platform Subaccount in SAP SuccessFactors

You need to register the assertion consumer service of SAP Cloud Platform subaccount as an authorized assertion consumer service in Provisioning of SAP SuccessFactors.

### Procedure

1. Download the service provider SAML metadata file from the SAP Cloud Platform cockpit.
  - a. Go to your subaccount and choose in the SAP Cloud Platform cockpit.
  - b. Choose *SAML Metadata* to download an XML file that contains the SAML 2.0 metadata describing SAP Cloud Platform as a service provider.
  - c. Open the XML file in a text editor and copy the following values:
    - o The value of the `Location` attribute of the `AssertionConsumerService` element with the `HTTP-POST` binding of the XML file: this is the value of the Assertion Consumer Service.
    - o The value of the `Location` attribute of the `SingleLogoutService` element with the `HTTP-POST` binding of the XML file: this is the value of the logout URL.
    - o The value of the `EntityID` attribute of `EntityDescriptor` element of the XML file: this is the value of the Audience URL.
2. In Provisioning of SAP SuccessFactors, go to your company and choose *Authorized SP Assertion Consumer Service Settings* under the *Service Provider Settings* section.
3. Choose *Add another Service Provider ACS* and fill in the following fields:

Field	Value
<i>Assertion Consumer Service</i>	The assertion consumer service URL  This is the value of the <code>Location</code> attribute of the <code>AssertionConsumerService</code> element with the <code>HTTP-POST</code> binding you copied in <b>step 1</b> .
<i>Logout URL</i>	The logout URL  This is the value of the <code>Location</code> attribute of the <code>SingleLogoutService</code> element with the <code>HTTP-POST</code> binding you copied in <b>step 1</b> .
<i>Audience Url</i>	The audience URL  This is the value of the <code>EntityID</code> attribute of <code>EntityDescriptor</code> element you copied in <b>step 1</b>

4. Choose *Save*.

## 4.2.2.2 Installing and Configuring SAP SuccessFactors Extension Applications for Cloud Foundry Environment

Use this procedure to configure the authentication of the applications with which you want to extend your SAP SuccessFactors systems and to install and configure them in SAP Cloud Platform, Cloud Foundry environment.

To use OAuth 2.0 for authentication, you will first need to register your OAuth client in the SAP SuccessFactors system, and set up the permissions required for this registration. Then, you can register your OAuth client application.

### Related Information

[Configuring Application Authentication \[page 728\]](#)

[Configuring the Extension Application's Connectivity to SAP SuccessFactors \[page 729\]](#)

### 4.2.2.2.1 Configuring Application Authentication

Use this procedure to configure the authentication of the applications with which you want to extend your SAP SuccessFactors systems.

### Prerequisites

You have the Cloud Foundry command line interface (cf CLI) downloaded and installed. See [Download and Install the Cloud Foundry Command Line Interface](#).

### Context

You need your own application router that connects your extension application to the centrally provided user account and authentication (UAA) service. This means that you need to create an application with an application router and configure it to route the requests to your extension application. Thus, the application router manages the user authentication for you.

#### i Note

For Node.js extension applications, you can optionally deploy the application router as part of your extension application.

## Procedure

1. Create an xsuaa service instance. See [Create a Service Instance from the xsuaa Service](#).
2. Bind the xsuaa service instance you have created to the extension application. See: [Bind the xsuaa Service Instance to the Application](#).
3. Create an application with an application router and configure it to forward the requests to your extension application. See [Configure Application Router](#).
4. Verify that there is an authentication for your extension application and access the application using the application router URL.
  - a. Get the URL of the application router using the `cf apps` command.
  - b. Enter the application router URL in a browser.
  - c. You are redirected to the SAP SuccessFactors identity provider that you have already configured. See [Establish Trust Between SAP SuccessFactors and SAP Cloud Platform \[page 705\]](#).
  - d. After a successful login, you will get redirected to the welcome page of your extension application if you have defined one.

## Related Information

[Authentication for Applications](#)

### 4.2.2.2 Configuring the Extension Application's Connectivity to SAP SuccessFactors

Use this procedure to configure the connectivity between your extension application and the SAP SuccessFactors system.

To be able to make calls to the SAP SuccessFactors OData APIs with user propagation, you need to create a destination with SAML Bearer Assertion authentication in the SAP Cloud Platform cockpit on a subaccount level. You also need to create an OAuth client in the SAP SuccessFactors system.

## Related Information

[Authentication for Java Resource Servers](#)

[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

## 4.2.2.2.2.1 Download the X509 Certificate in SAP Cloud Platform

### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your extension subaccount in the Cloud Foundry environment.
2. Choose  *Connectivity*  *Destinations*.
3. Choose *Download Trust* to get the certificate for this subaccount and save it on your local file system.
4. Open the certificate in a text editor and copy the content between `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`.

## 4.2.2.2.2 Create an OAuth Client in SAP SuccessFactors

### Procedure

1. In the SAP SuccessFactors system, go to *Admin Center* and search for **OAuth**. Choose *Manage OAuth2 Client Applications* from the search results.
2. Choose *Register Client Application*.
3. In the *Application Name*, choose a descriptive name for the client of your choice.
4. In the *Application URL* field, enter the URL of the extension application.
5. In the *X.509 Certificate* field, paste the content between `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` of the certificate you downloaded in the [Download the X509 Certificate in SAP Cloud Platform](#), step 4.
6. Choose *Register* to save the OAuth client.

### 4.2.2.2.3 Create an HTTP Destination in SAP Cloud Platform

You create an HTTP destination to be able to make calls to the SAP SuccessFactors OData APIs using SAML 2.0 Bearer Assertion authentication.

#### Procedure

1. In the SAP Cloud Platform cockpit, navigate to your extension subaccount in the Cloud Foundry environment.
2. Choose  *Connectivity Destinations*.
3. Choose *New Destination* and fill in the following properties:

Property	Value
<i>Name</i>	Enter a name for the destination. For example, <b>sap_hcmcloud_core_odata</b> .
<i>Type</i>	HTTP
<i>URL</i>	Enter the URL of the SAP SuccessFactors OData API you want to consume. For a list of the API Endpoint URL for the SAP SuccessFactors environments, see <a href="#">About HXM Suite OData APIs</a> .
<i>Proxy Type</i>	Internet
<i>Authentication</i>	OAuth2SAMLBearerAssertion
<i>Audience</i>	www.successfactors.com
<i>Client Key</i>	Enter the API Key of the OAuth client you created in SAP SuccessFactors.
<i>Token Service URL</i>	Enter the API Endpoint URL for the SAP SuccessFactors instance followed by <b>/oauth/token</b> . For example, <b>https://apisalesdemo2.successfactors.eu/oauth/token</b> . For a list of the API Endpoint URL for the SAP SuccessFactors environments, see <a href="#">About HXM Suite OData APIs</a> .
<i>System User</i>	The technical user for an OData access with SAML 2.0 Bearer Assertion authentication with technical user.

Property	Value
	Specify a value for this setting if you want to configure OData access with SAML 2.0 Bearer Assertion authentication with technical user.

4. In the *Additional Properties*, choose *New Property* to define the following properties:

Property	Value
<i>apiKey</i>	Enter the API Key of the OAuth client you created in SAP SuccessFactors.
<i>companyId</i>	The ID of your SAP SuccessFactors company.
<i>authnContextClassRef</i>	<code>urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession</code>
<i>nameIdFormat</i>	<code>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</code> if the user ID will be propagated to SAP SuccessFactors application or <code>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</code> if the user email will be propagated to SAP SuccessFactors.

5. Save the changes.

#### 4.2.2.2.4 Consume the Destination

### Context

To consume the destination you have created, you use the Destination service. You can either consume the Destination service directly, or configure the application router to consume it.

- 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 \(Cloud Foundry Environment\)](#).

## 4.2.3 Extending SAP Cloud for Customer in the Cloud Foundry Environment Manually

This section guides you through the configuration tasks that you need to perform to enable the SAP Cloud Platform, Cloud Foundry environment for developing extension applications for your SAP Cloud for Customer system.

### Process Flow

You have to configure the cloud platform extension integration with SAP Cloud for Customer to enable the use of applications running on top of the platform from SAP Cloud for Customer.

#### 1. Authentication and Principal Propagation

A single sign-on configuration between SAP Cloud for Customer and SAP Cloud Platform and the principal propagation enablement ensure the secure and consistent access for the extension solutions. If your solution requires system-to-system data access, with no user propagation, you can implement it using a user with Basic authentication and no principal propagation.



We recommend that you use SSO and OData access with principal propagation, to ensure that the data is accessed on behalf of the proper authorized user.

##### Implementation with SSO and principal propagation

When your scenario requires SSO and principal propagation, the SAP Cloud for Customer system and the SAP Cloud Platform subaccount (where the extension application is deployed or subscribed) have to trust each other and use one and the same identity provider. For more information, see [Configuring Single Sign-On on Cloud Foundry Environment \[page 734\]](#).

##### Implementation without SSO and principal propagation

If your scenario does not require SSO, principal propagation, or UI integration, you use a dedicated user and create an HTTP destination with Basic Authentication to configure the connectivity. For more information, see [Create and Configure the HTTP Destination \[page 750\]](#).

#### 2. Connectivity Between SAP Cloud Platform and SAP Cloud for Customer

When your extension application needs to do read and write operations from/to the SAP Cloud for Customer system, you have to configure the connectivity to enable the use of SAP Cloud for Customer OData APIs. You need such connectivity to be established also when you want to embed the UIs of the extension solution into the SAP Cloud for Customer user interface.

The connectivity configuration steps are required on both systems and can be implemented with the following options depending on the security mechanisms for protecting the connectivity:

##### Using OAuth



- On the SAP Cloud for Customer side:
  - [Configure OAuth Identity Provider in SAP Cloud for Customer \[page 749\]](#)
  - [Configure the OAuth Client for OData Access \[page 750\]](#)
- On the SAP Cloud Platform side:
  - Create an HTTP destination with OAuthSAMLBearerAssertion for authentication method. For more details, see [Create and Configure the HTTP Destination \[page 750\]](#).

### **Using Basic Authentication**

- On SAP Cloud for Customer side:
  - (If not available) Create a technical user with permissions to access the required SAP Cloud for Customer OData APIs and use it for the HTTP destination configuration when you choose Basic authentication for protecting the connectivity.
- On the SAP Cloud Platform side:
  - Create an HTTP destination with Basic authentication using the credentials of a technical user in the SAP Cloud for Customer system. For more details, see [Create and Configure the HTTP Destination \[page 750\]](#).

### **3. User Interface Integration**

The extension capabilities of the SAP Cloud Platform allow developers to embed the user interface of the new solution in the SAP Cloud for Customer screens and this way to offer seamless end-user experience.

The UI integration is via the HTML mashups of the SAP Cloud for Customer solution. For more details how to create an HTML mashup for your new extension solution and how to make it visible in the SAP Cloud for Customer screens, see [Embedding User Interface of an Extension Application in SAP Cloud for Customer](#).

### **4.2.3.1 Configuring Single Sign-On on Cloud Foundry Environment**

You can configure the cloud platform extension integration with SAP Cloud for Customer to enable the use of applications running on top of the platform from SAP Cloud for Customer.

### **Using Identity Authentication**

Identity Authentication service provides authentication, single sign-on, and on-premise integration. Identity Authentication service is closely integrated with SAP Cloud Platform, and it is offered as part of the platform.

To ensure the required security for accessing the extension applications, you need to configure the Single Sign-On between the SAP Cloud Platform extension subaccount and the SAP Cloud for Customer tenant using a SAML identity provider, for example SAP Cloud Platform Identity Authentication. The Single Sign-On requires both solutions to be configured as trusted SAML service providers for the Identity Authentication service and on the other side the Identity Authentication service to be configured as trusted identity provider for the two solutions.

In this scenario, the authentication to SAP Cloud for Customer extension applications is restricted to the authorized users. The identity of a user is verified by the identity provider, as specified by SAML 2.0. The

identity provider, (Identity Authentication), stores a list of all users that are allowed to access the service provider (SAP Cloud for Customer) along with their credentials. The integration between the SAP Cloud for Customer and the Identity Authentication is based on trust configuration. When a user attempts to access SAP Cloud for Customer 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 he or she, for example, tries to use the extension application. This is called Single Sign-On (SSO).

## Using Third-Party Identity Provider

You can use a third-party identity provider (which means a different from SAP Cloud Platform Identity Authentication) as well to ensure the required security for your landscape. In this case you also need to perform a few configuration tasks on all the sides - SAP Cloud Platform, SAP Cloud for Customer, and the identity provider that you are using.

### Related Information

[Configuring Single Sign-On Using Identity Authentication \[page 735\]](#)

[Configuring Single Sign-On Using Third-Party Identity Provider \[page 746\]](#)

### 4.2.3.1.1 Configuring Single Sign-On Using Identity Authentication

To ensure the required security for your landscape you need to perform a few configuration tasks on all the sides - SAP Cloud Platform, Identity Authentication and SAP Cloud for Customer.

### Context

The following procedure describes how to configure Identity Authentication and SAP Cloud for Customer to use the authentication and single sign-on capabilities based on the industry standard SAML 2.0. You can also find information about how to configure the trust between Identity Authentication and the SAP Cloud Platform subaccount on Cloud Foundry environment, in case it is not configured by default.

### Procedure

1. Configure the trust settings between Identity Authentication and your SAP Cloud for Customer system. For more information, see [Setting Up Trust Between Identity Authentication and SAP Cloud for Customer \[page 736\]](#).

### i Note

If you already have a SAP Cloud for Customer system with existing users, you need to define these users in the SAP Cloud Platform Identity Authentication service as well. To do that, export as a CSV file the users of your company employees from your SAP Cloud for Customer system and import it into Identity Authentication.

2. Configure the SAP Cloud Platform trust settings and add the tenant of Identity Authentication available for your company as a SAML identity provider. For more information, see [Setting Up Trust Between Identity Authentication and SAP Cloud Platform \[page 745\]](#).

## 4.2.3.1.1.1 Setting Up Trust Between Identity Authentication and SAP Cloud for Customer

To use SAP Cloud Platform Identity Authentication as a common identity provider between SAP Cloud Platform, Cloud Foundry environment and SAP Cloud for Customer, first you need to configure it for the SAP Cloud for Customer system.

### Procedure

1. Configure SAP Cloud Platform Identity Authentication.
2. Configure SAP Cloud for Customer.
3. Configure an application's Home URL.

## 4.2.3.1.1.1 Configuring Identity Authentication

Use this procedure to configure the service provider (SAP Cloud for Customer) in the SAP Cloud Platform Identity Authentication tenant and to define the identity federation.

### Procedure

1. Get the service provider metadata XML file. To do so, you download the metadata XML file of your SAP Cloud for Customer system:
  - a. Log on to your SAP Cloud for Customer system as an administrator.
  - b. Select  **ADMINISTRATOR**  **Common Tasks**  and then choose *Configure Single Sign-On*.
  - c. On the **CONFIGURE SINGLE SIGN-ON** screen, navigate to  **MY SYSTEM**  **GENERAL**  **Download Metadata** .
  - d. Choose the **SP Metadata** link, to download the SAP Cloud for Customer metadata XML file. Save it locally on your file system to import it later on in the Identity Authentication.

2. Create a custom application to use it as a SAML2.0 service provider. Access the tenant's administration console for Identity Authentication by using the console's URL.

The URL has the `https://<tenant ID>.accounts.ondemand.com/admin` pattern.

You can also get the URL from the Identity Authentication tenant registration e-mail.

- a. Choose from the menu on the left.
- b. Choose the `+Add` button on the left hand panel, and enter the name of your SAP Cloud for Customer system.

#### Note

The name of the application is displayed on the login and registration pages.

- c. Choose `Save`.

The system creates an application and adds it to the list of applications.

3. Configure the SAML 2.0 trust with SAP Cloud for Customer as a service provider. In the tenant's administration console for Identity Authentication, execute the following steps:
  - a. On the left-hand side, choose the newly created application, and then choose `Trust`.
  - b. Choose `SAML 2.0 Configuration`.
  - c. Upload the metadata XML file of your SAP Cloud for Customer system that you have downloaded in **Step 1**.

On service provider metadata upload, the fields are populated with the parsed data from the XML file.

- d. Save the configuration settings.

4. Configure the identity federation on Identity Authentication. To do so, proceed as follows:
  - a. In the tenant's administration console for Identity Authentication, choose your SAP Cloud for Customer application, and navigate to , and then select `Login Name`.

This is the profile attribute that Identity Authentication sends to the application as a name ID. The application then uses this attribute to identify the user. You should select the attribute expected by your SAP Cloud for Customer system as a valid user.

#### Note

E-mail is not supported by the SAP Cloud for Customer system.

- b. Save your selection.

## 4.2.3.1.1.2 Configuring SAP Cloud for Customer

Configure the Single Sign-On (SSO) to SAP Cloud Platform Identity Authentication in the SAP Cloud for Customer system.

### Procedure

1. Get the identity provider metadata XML file. To do so:
  - a. Use the following URL to access the metadata for the Identity Authentication tenant:  
`https://<tenant ID>.accounts.ondemand.com/saml2/metadata.`
  - b. Save the content of the page locally on your file system as an XML file.
2. Log on to your SAP Cloud for Customer system as an administrator.
3. Choose **ADMINISTRATOR** ➤ **Common Tasks** and then choose *Configure Single Sign-On*.
4. On the **CONFIGURE SINGLE SIGN-ON** screen, choose the **IDENTITY PROVIDER** tab.
5. Choose *New Identity Provider*.
6. Browse and open the metadata XML file that you have downloaded in **Step 1**. By importing the metadata, the system automatically uploads the required signature certificate and encryption certificate.  
The new identity provider is activated and displayed in the *Trusted Identity Provider* list.
7. Once you have configured your identity provider, activate SSO in your SAP Cloud for Customer system. To do so, choose *Activate Single Sign-On*, and then choose **OK**.
8. To save your settings, choose *Save* in the upper left-hand corner.

## 4.2.3.1.1.3 Configuring the Application's Home URL

You can configure the *Home URL* of an application in the administration console for SAP Cloud Platform Identity Authentication.

### Context

Home URL is the URL that the user is redirected to after being authenticated. Initially, the *Home URL* for an application is not configured in the administration console for Identity Authentication. Once the URL has been set, you can change it.

#### → Remember

*Home URL* is necessary when you import new users in Identity Authentication. Identity Authentication needs to send activation e-mails to the new users and the home URL has to be mentioned in the e-mails. To access the application, the users have to activate their user accounts. For more information see [Importing or Updating SAP Cloud for Customer Users in Identity Authentication \[page 742\]](#).

To configure the *Home URL*, proceed as follows:

## Procedure

1. Access the tenant's administration console for Identity Authentication by using the console's URL.

### i Note

The URL has the `https://<tenant ID>.accounts.ondemand.com/admin` pattern.

*Tenant ID* is an automatically generated ID by the system. The first administrator created for the tenant receives an activation e-mail with a URL in it. This URL contains the *tenant ID*.

2. Choose  *Applications & Resources*  from the menu on the left.

This operation opens a list of the applications.

3. Choose the application that you want to edit.

### i Note

Type the name of the application in the search field to filter the list items, or choose the application from the list on the left.

If you do not have a created application in your list, you can create one. For more information, see [Create a New Application](#).

4. Select the *Home URL* anchor text.

### i Note

You get the Home URL from the SAP Cloud for Customer system:

- Log on to your SAP Cloud for Customer system as an administrator.
- Choose  *ADMINISTRATOR*  and then choose *Configure Single Sign-On*.
- In the *Single Sign-On URL Handling* section, copy and save the URL from the *SSO URL* field. If this URL starts with uppercase *HTTPS* protocol, replace it with *https*.

- If you are configuring the URL for the first time, type the address in the pop-up dialog that appears.
- If you are editing the URL, choose *Edit* from the list item, and type the new address in the pop-up dialog.

5. Save your changes.

Once the application has been updated, the system displays the message `Application <name of application> updated`.

## 4.2.3.1.1.2 Uploading Users to Identity Authentication

When you configure the SAP Cloud Platform Identity Authentication service as the SAML identity provider for your SAP Cloud for Customer system, you have to make sure that all users from SAP Cloud for Customer will

have an identity record in the Identity Authentication service. For this purpose, you have to export the user details in a CSV file format and then use this CSV file to import these users into the Identity Authentication tenant of your company that will be used as the identity provider.

## Context

To upload the users of your SAP Cloud for Customer company into SAP Cloud Platform Identity Authentication, you export the user base and import it into SAP Cloud Platform Identity Authentication in CSV format.

## Procedure

1. Export the SAP Cloud for Customer users and save the data in CSV format. For more information, see [Exporting SAP Cloud for Customer Users \[page 740\]](#).
2. Import the users (user names and user IDs) in the Identity Authentication tenant with the CSV file you have exported from your SAP Cloud for Customer system. For more information, see [Importing or Updating SAP Cloud for Customer Users in Identity Authentication \[page 742\]](#).

### 4.2.3.1.1.2.1 Exporting SAP Cloud for Customer Users

Use this procedure to export SAP Cloud for Customer users and save the user data in CSV format.

## Context

You need to export the users from the SAP Cloud for Customer system, adapt the data and save it in CSV format so that you can import the data in the SAP Cloud Platform Identity Authentication tenant and thus allow your company employees to authenticate with their corporate credentials.

#### i Note

It is mandatory to have an e-mail for every user that you replicate from the SAP Cloud for Customer system in your Identity Authentication tenant.

Although in the SAP Cloud for Customer system having an e-mail for a user is not obligatory, you need to have this e-mail as a parameter when you export the users and import them in your SAP Cloud Platform subaccount.

The CSV file with the SAP Cloud for Customer user data must contain the following columns:

- status
- loginName
- mail

- `firstName`
- `lastName`

## Procedure

1. Log on to your SAP Cloud for Customer system as an administrator.
2. Open the Business Users view. To do so, choose **ADMINISTRATOR** **GENERAL SETTINGS** **Business Users**.
3. To export the users, choose **Export** **To Microsoft Excel®**.

The system exports the users in a Microsoft Excel® spreadsheet. The spreadsheet contains the following columns: *User Locked*, *Password Locked*, *User ID*, *Name*, *Technical ID*. The spreadsheet does not contain a column with the user e-mails and the user status.

4. Modify the spreadsheet as follows:
  - a. Remove the superfluous data and leave only the *User ID* and *Name* columns.
  - b. Rename the *User ID* column to **`loginName`** and split the *Name* column to **`firstName`** and **`lastName`** columns.

The data in the *loginName* stays the same, while the names from the former *Name* columns are split in the following way: first names go to the *firstName* column and last names go to the *lastName* column.

- c. Insert a new column before the *loginName* column and name it **`status`**. The status can be **`active`** or **`inactive`**.
- d. Insert a new column between *loginName* and *firstName* and name it **`mail`**.

Go to the SAP Cloud for Customer system. To get the e-mail of a user, select the user. The e-mail is displayed on the *General Data* tab page.

### i Note

- The *status*, *loginName*, *mail* and *firstName* columns must be with a string value of up to 32 characters. The *lastName* column must be with a string value of up to 64 characters.
- The names in the *mail* and *loginName* columns must be unique.
- You cannot change the e-mail of an existing user.
- The *status* column defines whether the user is still active in the system and is able to work with any tenant applications. When a user is deleted, it is rendered inactive.

- e. Save the file in CSV format.

After you have modified and saved your file, you should have a CSV file with the following columns and similar data.

<b><code>status</code></b>	<b><code>loginName</code></b>	<b><code>mail</code></b>	<b><code>firstName</code></b>	<b><code>lastName</code></b>
active	EID00001	michael.adams@example.com	Michael	Adams

status	loginName	mail	firstName	lastName
active	EID00002	julie.armstrong@example.com	Julie	Armstrong
active	EID00003	donna.moore@example.com	Donna	Moore

### 4.2.3.1.1.2.2 Importing or Updating SAP Cloud for Customer Users in Identity Authentication

As a tenant administrator of the SAP Cloud Platform Identity Authentication, you can import new users or update existing ones for a specific application with a CSV file, and send activation e-mails to the users that have not received activation e-mails for that application so far.

#### Prerequisites

- You are assigned the *Manage Applications* and *Manage Users* roles. For more information about how to assign administrator roles, see [Edit Administrator Authorizations](#).
- You have configured the trust between Identity Authentication tenant and the SAP Cloud for Customer system. For more information see [Setting Up Trust Between Identity Authentication and SAP Cloud for Customer \[page 736\]](#).

#### i Note

You need the metadata to configure the trust between the service provider and SAP Cloud Platform Identity Authentication, which is in the role of identity provider.

#### Context

By importing new users with a CSV file, you create user profiles without passwords in SAP Cloud Platform Identity Authentication. As a result, the users receive e-mails with instructions how to activate their user accounts. After the users set their passwords, they can log on to the application for which they were imported. Based on the user access configuration of the application, the users can log on to other applications connected with the tenant in Identity Authentication.

#### i Note

When a new application is created in the Identity Authentication tenant, the default value for user access is internal and you have to keep it like this. If you decide to change the user access to private for your SAP Cloud Platform subaccount or for the SAP Cloud for Customer system, you have to make sure you import the users into that application. For more details, see

In addition to the new user import, you can specify existing users in the imported CSV file. You thus define the users to be updated in Identity Authentication.

The CSV file contains these columns *status*, *loginName*, *mail*, *firstName*, *lastName*. These columns are mandatory and they must always have values.

The *status*, *loginName*, *mail* and *firstName* columns must be with a string value of up to 32 characters. The *lastName* column must be with a string value of up to 64 characters.

The names in the *mail* and *loginName* columns must be unique.

#### ⚠ Caution

You cannot change the e-mail of an existing user.

The *status* column defines whether the user is still active in the system and is able to work with any tenant applications. When a user is deleted, it is rendered inactive.

#### ❖ Example

status	loginName	mail	firstName	lastName
active	EID00001	michael.adams@example.com	Michael	Adams
active	EID00002	julie.armstrong@example.com	Julie	Armstrong
active	EID00003	donna.moore@example.com	Donna	Moore

To import users for an application into Identity Authentication, and to send activation e-mails, proceed as follows:

## Procedure

1. Access the tenant's administration console for Identity Authentication by using the console's URL.

#### ℹ Note

The URL has the `https://<tenant ID>.subaccounts.ondemand.com/admin` pattern.

*Tenant ID* is an automatically generated ID by the system. The first administrator created for the tenant receives an activation e-mail with a URL in it. This URL contains the *tenant ID*.

2. Choose the *Import Users* tile.

This operation opens the *Import Users* page.

3. Choose the application that you want to edit. This is the application you created in [Setting Up Trust Between Identity Authentication and SAP Cloud for Customer \[page 736\]](#).

### i Note

Type the name of the application in the search field to filter the list items, or choose the application from the list on the left.

If you do not have a created application in your list, you can create one. For more information, see [Setting Up Trust Between Identity Authentication and SAP Cloud for Customer \[page 736\]](#).

4. Choose the *Browse...* button and specify the location of the CSV file.

### i Note

Use a file smaller than 100 KB and with an extension .csv. If your file is 100 KB or larger, you have to import the user information in iterations with smaller size files.

5. Choose the *Import* button.

If the operation is successful, the system displays the message `Users imported or updated`.

6. Choose the one of the following options:

Option	Description
<b>Do nothing</b>	The users are imported or updated for the selected application, but they will not receive activation e-mails. The activation e-mails will be sent when you choose ► <i>Send E-Mails</i> ► <i>Send</i> ▶.
<b>Repeat steps 2 to 5</b>	The users are imported or updated for the selected application, but they will not receive activation e-mails. The activation e-mails will be sent when you choose ► <i>Send E-Mails</i> ► <i>Send</i> ▶.
<b>Choose ► <i>Send E-Mails</i> ► <i>Send</i> ▶</b>	This will send activation e-mails to all users that are imported for the selected application, but have not received activation e-mails so far.



### i Note

The *Send* button is inactive if *Home URL* or SAML 2.0 configuration of the application is missing. You can only import users, but you cannot send activation emails.

You need the *Home URL* configured for the specific application to be able to send the activation e-mails to the imported new users. For more information, see [Configuring the Application's Home URL \[page 738\]](#).

To access the application, the users have to activate their user accounts by following the link they receive in the e-mails.

### 4.2.3.1.1.3 Setting Up Trust Between Identity Authentication and SAP Cloud Platform

Use this procedure to configure the SAP Cloud Platform, Cloud Foundry environment trust settings and to add the tenant of SAP Cloud Platform Identity Authentication registered for your current SAP user as an identity provider.

#### Context

Identity Authentication is closely integrated with SAP Cloud Platform, and it is offered as part of most of the cloud platform packages. For those packages the trust between the SAP Cloud Platform subaccount and Identity Authentication is configured automatically and the tenant for Identity Authentication is set up by default, once you have a partner or customer subaccount. However, you can manually configure the trust and set up the Identity Authentication tenant if your scenario requires it.

#### Procedure

1. In the Identity Authentication tenant, access the administration console by using the console's URL. You need to use another browser, or incognito session of the same browser.  
The URL has the `https://<tenant ID>.accounts.ondemand.com/admin` pattern.  
You can also get the URL from the Identity Authentication tenant registration e-mail.
2. You have to save the metadata of your Identity Authentication tenant on your local file system as an XML file. You can either find the tenant at `https://<tenant ID>.accounts.ondemand.com/saml2/metadata` or access it via [Applications & Resources](#) [Tenant Settings](#) [SAML 2.0 Configuration](#). Then choose the [Download Metadata File](#) link. You will need this metadata in **Step 5**.
3. Open the SAP Cloud Platform cockpit and select the region in which your subaccount is hosted. Select the global account that contains your subaccount, and then choose the tile of your subaccount. For more information about regions, see [Regions and Hosts](#).
4. Choose [Security](#) [Trust Configuration](#).
5. Choose [New Trust Configuration](#) and fill in the required fields:
  - a. In the [Metadata](#) field, upload the metadata file you have downloaded from the Identity Authentication tenant in **step 2**. All the required fields are automatically filled in.
  - b. In the [Name](#) field, enter a meaningful name of your choice.
  - c. Choose [Save](#).
6. In the SAP Cloud Platform cockpit, download the service provider SAML metadata file. Open the link `https://<subaccount_subdomain>.authentication.<region_host>/saml/metadata` where:
  - `<subdomain>` is part of the SAP Cloud Platform subaccount details in the cockpit.
  - `<region_host>` is the API endpoint without `api.cf`. See [Regions and Hosts](#).

When you are prompted, save the XML file on your local file system. This file contains the SAML 2.0 metadata describing SAP Cloud Platform as a service provider.

You will need this metadata in **Step 11**.

7. In the Identity Authentication tenant, access the administration console by using the console's URL. You need to use another browser, or incognito session of the same browser.

The URL has the `https://<tenant ID>.accounts.ondemand.com/admin` pattern.

You can also get the URL from the Identity Authentication tenant registration e-mail.

8. Choose  .
9. Choose the `+Add` button on the left-hand panel, and enter the name of your SAP Cloud Platform subaccount.
10. Choose `Save`.
11. Configure the SAML 2.0 trust with SAP Cloud Platform subaccount as a service provider. To do so, proceed as follows:
  - a. On the left-hand side, choose the newly created application, and then choose `Trust`.
  - b. Choose `SAML 2.0 Configuration`.
  - c. Upload the metadata XML file of your SAP Cloud Platform subaccount that you have downloaded in **Step 6**.

On service provider metadata upload, the fields are populated with the parsed data from the XML file.

- d. Save the configuration settings.

## Results

The trust will be established automatically upon registration on both the SAP Cloud Platform and Identity Authentication tenant side.

## Related Information

[Get Started with Identity Authentication](#)

[ID Federation with a Identity Authentication Tenant](#)

### 4.2.3.1.2 Configuring Single Sign-On Using Third-Party Identity Provider

To ensure the required security for your landscape you need to perform a few configuration tasks on all the sides - SAP Cloud Platform, SAP Cloud for Customer, and the identity provider that you are using (if this provider is different from SAP Cloud Platform Identity Authentication, for which there is a dedicated section).

## Procedure

1. Set up trust between your identity provider and SAP Cloud for Customer system.

For more information, see section **Configure Your Solution for Single Sign-On** in the SAP Cloud for Customer Security Guide at <https://service.sap.com/%7Esapidb/012002523100016515992016E/C4CSecurityGuide1611.pdf>.

2. Set up trust between your identity provider and SAP Cloud Platform.

For more information, see [Identity Federation](#).

### 4.2.3.2 Configuring the Extension Application

When configuring the SAP Cloud Platform extension application and you have SAP Cloud Platform subaccount in the Cloud Foundry environment, you have to set up the connectivity from the SAP Cloud Platform subaccount to the SAP Cloud for Customer system..

#### Related Information

[Configure Application Authentication \[page 747\]](#)

[Configuring the SAP Cloud Platform Application Connectivity to SAP Cloud for Customer APIs \[page 748\]](#)

#### 4.2.3.2.1 Configure Application Authentication

#### Prerequisites

- Have the Cloud Foundry command line interface (cf CLI) downloaded and installed. See [Download and Install the Cloud Foundry Command Line Interface](#).
- Have Node.JS including its NPM Packager Manager installed. See <https://nodejs.org/en/>.

#### Context

You need your own application router that connects your extension application to the centrally provided user account and authentication (UAA) service. This means that you need to deploy an approuter as part of your application that manages the user authentication for you.

The approuter has these main functions:

- Handles authentication for all apps of the application
- Serves static resources
- Performs route mapping (URL mapping)
- In case of multitenancy, it derives the tenant information from the URL and provides it to the extended services for User Account and Authentication service (XSUAA) to redirect the authentication request to the tenant-specific identity provider. See [User Account and Authentication Service of the Cloud Foundry Environment](#).

## Procedure

1. To set up the application router as part of your application, execute the following command in cf CLI:

```
npm config set @sap:registry=https://npm.sap.com
```
2. Create an xsuaa service instance. See [Create a Service Instance from the xsuaa Service](#).
3. Add an application router. See [Application Router and Destination](#)
4. Update the extension application by using the cf push command in the cf CLI.
5. Verify that there is an authentication for your application and access the application using the approuter URL.
  - a. Get the url of the approuter using the cf apps command.
  - b. Enter the approuter URL in a browser.
  - c. You are redirected to the Identity Authentication service that you have already configured. See [Setting Up Trust Between Identity Authentication and SAP Cloud for Customer \[page 736\]](#).
  - d. After a successful login, you will get redirected to the welcome page of your extension application if you have defined one.

### 4.2.3.2.2 Configuring the SAP Cloud Platform Application Connectivity to SAP Cloud for Customer APIs

To set up the connectivity from an SAP Cloud Platform subaccount to an SAP Cloud for Customer system, you need to create HTTP destinations in the SAP Cloud Platform cockpit. These destinations provide data communication via HTTP protocol.

You can use the SAML Bearer assertion flow for consuming OAuth-protected resources. Users are authenticated by using SAML against the configured trusted identity providers. The SAML assertion is then used to request an access token from an OAuth authorization server. This access token must be added as an Authorization header with value `Bearer <access token>` in all HTTP requests to the OAuth-protected resources.

These are the steps you need to follow:

1. [Configure OAuth Identity Provider in SAP Cloud for Customer \[page 749\]](#)
2. [Configure the OAuth Client for OData Access \[page 750\]](#)
3. [Create and Configure the HTTP Destination \[page 750\]](#)

## Related Information

[Authentication for Java Resource Servers](#)  
[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

### 4.2.3.2.2.1 Configure OAuth Identity Provider in SAP Cloud for Customer

You need to add the SAP Cloud Platform service provider as a trusted OAuth identity provider.

#### Procedure

1. Log on to the SAP Cloud Platform cockpit, and configure the settings as follows:
  - a. Navigate to your extension subaccount in the Cloud Foundry environment.
  - b. Choose [Connectivity](#) [Destinations](#).
  - c. Choose [Download Trust](#) to get the certificate for this subaccount and save it on your local file system.
  - d. Open the certificate in a text editor, copy the content between `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`, and save it in a file with the following format `<subaccount>_signing.cer`, where `<subaccount>` is the [Subaccount Name](#) from the [Subaccount Information](#) of your SAP Cloud Platform subaccount.
2. Log on to your SAP Cloud for Customer system as an administrator. Go to [ADMINISTRATOR](#) [Common Tasks](#). Choose [Configure OAuth 2.0 Identity Provider](#) [New OAuth2.0 Provider](#), and configure the settings as follows:
  - o In the [Issuing Entity Name](#) field, enter `cfapps.<region_host>/<subaccountID>`. You go to the SAP Cloud Platform cockpit, navigate to the subaccount, go to [Overview](#) and:
    - o For the `<region_host>`, copy the API endpoint from the [Cloud Foundry](#) section, and remove the `https://api.cf..`
    - o For the `<subaccountID>`, copy the ID from the [Subaccount Details](#) section.For example, `cfapps.eu10.hana.ondemand.com/12345678-1234-1234-1234-1234578`.
  - o From the [Primary Signing Certificate](#) field, browse the `<subaccount>_signing.cer` file that you saved on [step 1d](#).
  - o Select the [E-Mail Address](#) checkbox.
3. Choose [Submit](#).

## 4.2.3.2.2.2 Configure the OAuth Client for OData Access

In SAP Cloud for Customer, use this procedure to configure the OAuth client for OData access to SAP Cloud for Customer OData APIs.

### Procedure

1. Log on to your SAP Cloud for Customer system as an administrator. Go to  [ADMINISTRATOR](#)  [OAUTH2.0 CLIENT REGISTRATION](#) .
2. Choose [New](#).
3. Specify a client secret, description, and token lifetime (in seconds). For example, 3600 seconds.
4. In the *Issuer Name* field, use the dropdown list to specify the identity provider that you created in the document [Configure OAuth Identity Provider in SAP Cloud for Customer \[page 749\]](#).
5. Copy the entry in the *Client ID* field. You will need it later when creating the HTTP destination with OAuth authentication required for the connectivity to the SAP Cloud for Customer OData APIs.
6. In the *Scope* list, select the [UIWC:CC\\_HOME](#) scope.
7. Choose [Save and Close](#).

### Next Steps

On the SAP Cloud Platform side configure the HTTP destination required to create an HTTP client for the OData API, and thus ensure the connectivity to SAP Cloud for Customer. For more information, see [Create and Configure the HTTP Destination \[page 750\]](#).

### Related Information

[Configuring OAuth 2.0](#)

## 4.2.3.2.2.3 Create and Configure the HTTP Destination

### Prerequisites

You have logged into the SAP Cloud Platform cockpit from the SAP Cloud Platform landing page for your subaccount.

## Procedure

1. In the cockpit, go to the *Subaccounts* drop-down menu and choose your subaccount.
2. Choose   in the navigation panel.
3. Create an HTTP destination.

To enable principal propagation, create an *OAuth2SAMLBearerAssertion* HTTP destination and configure its settings as follows:

1. Configure the basic settings:

Parameter	Value
Name	Enter a meaningful name.
Type	<b>HTTP</b>
Description	(Optional) Enter a meaningful description.
URL	<code>https://&lt;my_SAP_Cloud_for_Customer_system_name&gt;.crm.ondemand.com/sap/c4c/odata/v1/c4codata</code>
Proxy Type	<b>Internet</b>
Authentication	<b>OAuth2SAMLBearerAssertion</b>
Audience	Go to the SAP Cloud for Customer administration view, then navigate to <i>Configure Single Sign-On</i> under <i>General Settings</i> and copy the value from the <i>Local Service Provider</i> field.
Client Key	Client ID Paste the entry you have copied from the <i>Client ID</i> field when configuring the OAuth client. For more information, see <a href="#">Configure the OAuth Client for OData Access [page 750]</a> .
Token Service URL	<code>https://&lt;my_SAP_Cloud_for_Customer_system_name&gt;.crm.ondemand.com/sap/bc/sec/oauth2/token</code>
Token Service User	Client ID Paste the entry you have copied from the <i>Client ID</i> field when configuring the OAuth client. For more information, see <a href="#">Configure the OAuth Client for OData Access [page 750]</a> .

Parameter	Value
Token Service Password	Client secret Paste the entry you have copied from the <a href="#">Client Secret</a> field when configuring the OAuth client. For more information, see <a href="#">Configure the OAuth Client for OData Access [page 750]</a> .

2. Configure the required additional property. To do so, in the *Additional Properties* panel, choose [New Property](#), and enter the following property:

**i Note**

You map the application users with the respective users in SAP Cloud for Customer using their email. Thanks to this mapping, you don't necessarily need to have a common identity provider between SAP Cloud Platform and SAP Cloud for Customer.

Parameter	Value
authnContextClassRef	<b>urn:None</b>
nameIdFormat	<b>urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</b>
scope	Scope ID entries separated by space. Paste the entry you have copied from the Client ID field when configuring the OAuth client. For more information, see <a href="#">Configure the OAuth Client for OData Access [page 750]</a> . Example: <i>UIWC:CC_HOME</i>
userIdSource	email

3. Select the [Use default JDK truststore](#) checkbox.

4. Save your entries.

## Related Information

[Consuming the Destination Service \(Cloud Foundry Environment\)](#)

# 5 Administration and Operations

Learn how to manage and configure global accounts and subaccounts, as well as how to operate your applications in the different environments.

- [Administration and Operations in the Cloud Foundry Environment \[page 916\]](#)
- [Administration and Operations in the ABAP Environment \[page 993\]](#)
- [Administration and Operations, Neo Environment](#)
- [Administration and Operations in the Kyma Environment \[page 1079\]](#)

## Related Information

[Account Model](#)

## 5.1 Account Administration

Learn how to manage global accounts, directories (beta), and subaccounts on SAP Cloud Platform using different tools.

- [Account Administration in the Cockpit \[page 754\]](#)
- [Account Administration Using the CLI for SAP Cloud Platform \(sapcp CLI\) \[Feature Set B\] \[page 850\]](#)

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 \[page 916\]](#)
- [Administration and Operations in the ABAP Environment \[page 993\]](#)
- [Administration and Operations in the Kyma Environment \[page 1079\]](#)

## Related Information

[Account Model](#)

[Entitlements and Quotas](#)

[User and Member Management](#)

[Environments](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1 Account Administration in the Cockpit

Learn about frequent administrative tasks you can perform using the SAP Cloud Platform cockpit, such as managing global accounts, subaccounts, entitlements and members.

- [Managing Global Accounts and Subaccounts Using the Cockpit \[page 754\]](#)
- [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#)
- [Security Administration: Managing Authentication and Authorization \[page 784\]](#)
- [Managing Resource Providers \[page 833\]](#)

### Related Information

[Account Model](#)

### 5.1.1.1 Managing Global Accounts and Subaccounts Using the Cockpit

Your SAP Cloud Platform global account is the entry point for managing the resources, landscape, and entitlements for your departments and projects in a self-service manner.

Set up your account model by creating subaccounts in your enterprise account. You can create any number of subaccounts in any environment (Neo, Cloud Foundry, ABAP, and Kyma) and region.

### Related Information

[Setting Up Your Account Model](#)

[Using Subaccounts to Create a Staged Development Environment](#)

### 5.1.1.1.1 Log On to Your Global Account [Feature Set A]

Use the cockpit to log on to your global account and start working in SAP Cloud Platform.

### Prerequisites

You have received a welcome e-mail from SAP for your global account.

## Context

### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

When your organization signs a contract for SAP Cloud Platform services, an e-mail is sent to the e-mail address specified in the contract. The e-mail message contains the link to log on to the system and the SAP Cloud Identity credentials (user ID) for the specified user. These credentials can also be used for sites such as the [SAP Store](#) or the [SAP Community](#).

## Procedure

1. Use the link in your Welcome e-mail.

Alternatively, for example, choose <https://account.eu1.hana.ondemand.com>. To avoid latency, make sure you choose a logon URL in the region closest to you.

### i 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.

### → Tip

In the Global Accounts page, you can filter the display of global accounts:

- Filter by role to display global accounts according to your role in the global account, administrator or member.
- Filter by region to display global accounts that contain subaccounts in the selected region.
- Filter by environment to display global accounts that contain subaccounts in the selected environment.

## Results

The global account [Overview](#) page 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. Open the *Subaccounts* page directly from the *Global Account Info* view or from the navigation pane.  
See [Create a Subaccount \[Feature Set A\] \[page 762\]](#).

### 5.1.1.1.2 Navigate to Global Accounts and Subaccounts [Feature Set A]

Learn how to navigate to your global accounts and subaccounts in the SAP Cloud Platform cockpit.

#### Prerequisites

##### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

- Sign up for an enterprise or a trial account and receive your logon data. For more information, see [Get a Trial Account \[page 7\]](#) or [Purchase a Customer Account \[page 8\]](#).
- Create the subaccount to which you want to navigate. For more information, see [Create a Subaccount \[Feature Set A\] \[page 762\]](#).

#### Procedure

Entity You'd Like to Navigate to	Navigation Path
Global Account	<ol style="list-style-type: none"> <li>1. Use the link in your Welcome e-mail or go to <a href="https://account.hana.ondemand.com">https://account.hana.ondemand.com</a>. 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.</li> <li>2. Choose a global account.</li> </ol>

Entity You'd Like to Navigate to	Navigation Path
<b>Result:</b>	You should see the following path in the breadcrumbs: Home /  <global_account>
<b>Subaccount</b>	<ol style="list-style-type: none"> <li>Select the global account that contains the subaccount you'd like to navigate to by following the steps described above.</li> <li>Select the subaccount.</li> </ol>

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.1.3 Navigate to Global Accounts and Subaccounts [Feature Set B]

Learn how to navigate to your global accounts and subaccounts in the SAP Cloud Platform cockpit.

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

## Navigate to Global Accounts

### Option 1 - You are only part of one global account

When you log on to the cockpit you are automatically taken into your global account, to the *Subaccounts* page.

### Option 2 - You are part of multiple global accounts

When you log on to the cockpit, a dialog opens up containing the list of all global accounts that you are 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 will 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 *Subaccounts* 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 will be saved and you will be 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 will look like this:

 <global account name>

You can navigate to a different global account by following these steps:

1. Navigate to the *Subaccounts* page at global account level (if you're not already in it).
2. Choose *Switch Global Account*.  
This opens up a dialog with a list of all global accounts that you are part of.
3. In the dialog, select the global account you want to navigate to and choose *Continue* to enter the desired global account.

You can tell that you are now in a different global account by looking at the breadcrumbs (see above).

## Navigate to Subaccounts

When you enter your global account, you are by default taken to the *Subaccounts* page of that global account. To navigate to a subaccount, simply choose the corresponding tile from this page.

Once you've entered a subaccount, the breadcrumbs will look like this:

 <global account name> /  <subaccount name>

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.4 Change the Display Name of Your Global Account [Feature Set A]

Change the display name for the global account using the SAP Cloud Platform cockpit.

### Prerequisites

You are a member of the global account that you'd like to edit.

### Context

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

The overview of global accounts available to you is your starting point for viewing and changing global account details in the cockpit. To view or change the display name for a global account, trigger the intended action directly from the tile for the relevant global account.

### Procedure

1. Choose the global account for which you'd like to change the display name and choose  on its tile.  
A new dialog shows up with the mandatory *Display Name* field that is to be changed.
2. Enter the new human-readable name for the global account.
3. Save your changes.

### Related Information

[Account Model](#)

## 5.1.1.5 Add Members to Your Global Account [Feature Set A]

Add users as global account members using the SAP Cloud Platform cockpit.

### Prerequisites

You have the Administrator role in the global account.

### Context

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

The users you add as members at global account level are automatically assigned the Administrator role.

All global account administrators can do the following:

- View all the subaccounts in the global account, meaning all the subaccount tiles in the global account's [Subaccounts](#) page.
- Edit general properties of the subaccounts in the global account from the [Edit](#) icon in the subaccount tile.
- Create a new subaccount in the global account.
- View, add, and remove global account members.
- Manage entitlements for the global account.

#### ! Restriction

Adding members to global accounts is only possible in enterprise accounts, not in trial accounts.

On the [Members](#) page at the global account level in the cockpit, all global account members can view the global account administrators.

### Procedure

1. Choose the global account to which you'd like to add members.
2. In the navigation area, choose [Members](#).
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses, separated by commas, spaces, semicolons, or line breaks and choose [Add Members](#).

## Next Steps

To delete members at global account level, choose  ([Delete](#)) next to the user's ID.

## Related Information

[User and Member Management](#)  
[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.1.6 Add Members to Your Global Account [Feature Set B]

Add members to your global account by assigning them a predefined role collection.

## Prerequisites

You must be a global account administrator in order to add other global account members.

## Context

### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

There are 2 predefined role collections that you can use when adding global account members:

- Global Account Administrator
- Global Account Viewer

For more information on the roles included in each of these role collections, see [Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#).

## Procedure

1. In your global account, use the side navigation to go to   [Security](#) > [Trust Configuration](#).

2. Choose the identity provider.
3. Enter the email address of the platform user you want as a global account member and choose *Show Assignments*.
4. **Optional:** If the user is not part of the identity provider choose *Add User* in the dialog that appears.
5. **Optional:** Choose *Assign Role Collection*, select the role collection you want to assign to the user and choose *Assign Role Collection* once more in the dialog to confirm.

## Results

The new role collection assigned to the user is displayed in the table. The user is now a global account member.

## Related Information

[User and Member Management](#)  
[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.7 Create a Subaccount [Feature Set A]

Create subaccounts in your global account using the SAP Cloud Platform cockpit.

## Prerequisites

You are a global account administrator.

### → Recommendation

Before creating your subaccounts, we recommend you learn more about [Setting Up Your Account Model](#).

## Context

### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

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.

Running on the cloud management tools feature set A: You can enable subaccounts to use beta features, including services and applications, which are occasionally made available by SAP for SAP Cloud Platform. This option is not selected by default and available only to administrators for your enterprise account.

#### ⚠ Caution

You shouldn't use SAP Cloud Platform beta features in subaccounts that belong to productive enterprise accounts. Any use of beta functionality is at the customer's own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

## Procedure

1. From your global account, choose [New Subaccount](#).
2. Specify a display name.
3. **Optional:** Specify a description.
4. Leave the [Neo Environment](#) checkbox deselected and choose the desired infrastructure provider and region for your subaccount.
5. Enter a subdomain for your subaccount. This will become part of the URL for accessing applications that you subscribe to from this subaccount.

#### ℹ Note

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 (e.g. **SUBDOMAIN** and **subdomain** are considered to be the same).

6. If your subaccount is to be used for production purposes, select the [Used for production](#) option.

By flagging your subaccount with this option, you can help SAP Support to take appropriate action when handling incidents that are related to mission-critical accounts in production systems.

#### ℹ Note

Do not select this option if your subaccount is intended for non-production purposes, such as development, testing, and demos. You can change your selection at any time by editing the subaccount properties.

7. **Optional:** To use beta services and applications in the subaccount, select [Enable beta features](#).
8. Save your changes.

## Results

A new tile appears in the global account page with the subaccount details.

## Next Steps

Once you've created your subaccount, navigate to it to enable the environment that you wish to use.

## Related Information

### Global Accounts

[Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

### Using Beta Features with Subaccounts

[Org Administration Using the Cockpit \[page 916\]](#)

[Relationship Between Global Accounts and Subaccounts \[Feature Set A\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.1.8 Create a Subaccount [Feature Set B]

Create subaccounts in your global account using the SAP Cloud Platform cockpit.

### Prerequisites

You're a global account administrator.

### Context

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

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're automatically assigned to it as a security administrator.

### Procedure

1. From your global account, choose [New Subaccount](#).
2. Specify a display name.

3. **Optional:** Specify a description.
4. Choose the infrastructure provider and region for your subaccount.
5. Enter a subdomain for your subaccount. This will become part of the URL for accessing applications that you subscribe to from this subaccount.

**i Note**

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 doesn't qualify as a means to differentiate subdomains (for example, **SUBDOMAIN** and **subdomain** are considered to be the same).

6. **Optional:** Assign custom properties to the subaccount to make organizing and filtering your subaccounts easier.
7. If your subaccount is to be used for production purposes, select the *Used for production* option.

When you flag your subaccount with this option, you can help your cloud operator to take appropriate action when handling incidents that are related to mission-critical accounts in production systems. Selecting this option doesn't change your subaccount in any way.

**i Note**

Don't select this option if your subaccount is intended for nonproduction purposes, such as development, testing, and demos. You can change your selection at any time by editing the subaccount properties.

8. Save your changes.

## Results

A new tile appears in the global account page with the subaccount details.

## Next Steps

Once you've created your subaccount, navigate to it to enable the environment that you wish to use.

## Related Information

### Global Accounts

[Navigate to Global Accounts and Subaccounts \[Feature Set B\] \[page 757\]](#)

[Org Administration Using the Cockpit \[page 916\]](#)

[Relationship Between Global Accounts and Subaccounts \[Feature Set A\]](#)

[Custom Properties \[Feature Set B\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.9 Managing Security Administrators in Your Subaccount [Feature Set A]

Running on the cloud management tools feature set A: When you create a subaccount, SAP Cloud Platform automatically grants your user the role for the administration of business users and their authorizations in the subaccount. Having this role, you can also add or remove other users who will then also be user and role administrators of this subaccount.

After having created a subaccount in the Cloud Foundry environment of SAP Cloud Platform, your user automatically has the administration role. This means that your user also has the [Security](#) tab, where you can perform security administration tasks. As a security administrator, you can manage authentication and authorization in this subaccount, such as configuring trust to identity providers, and assigning role collections to business users.

You can delegate the security administration to other users. Simply add the users as security administrators to your subaccount. SAP Cloud Platform grants the `User & Role Administrator` role to these users.

To see the `User & Role Administrator` role and all users with this role, go to your subaccount (see [Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)). Choose  [Security](#)  [Administrators](#).

You can also remove the users who are not supposed to have this role.

### i Note

All users with the `User & Role Administrator` role can manage this subaccount, including the security administration tasks.

## Related Information

[User and Member Management](#)  
[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.9.1 Add Security Administrators to Your Subaccount Feature Set A

Running on the cloud management tools feature set A: You can add security administrators by adding users to your subaccount and by assigning the corresponding role to them.

## Prerequisites

- You can add users as security administrators: You either created this subaccount, or an authorized user has assigned the `User & Role Administrator` role to your user for it. The users can come from

different identity providers. The origin uniquely identifies the identity provider that stores this user. If your user is stored in SAP ID service, the user's origin is *sap.ids*. If your user is stored in a custom platform identity provider, the user has a different origin.

- Your user and the users you want to add must have at least one of the following member roles (see the related link):
  - 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.

## Context

To add security administrators to your subaccount, take the following steps.

## Procedure

1. Open the SAP Cloud Platform cockpit.
  2. Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)). Since your user is the security administrator of your subaccount, you see *Security* in the navigation area.
  3. Choose  *Security* > *Administrators* 
  4. To add security administrators to your subaccount, choose *Add Administrators*.
- A popup opens, where you can enter the user ID and the origin and assign roles.
5. Enter the user IDs of the users you want to add as security administrators.

### Note

Use the e-mail address as user ID.

6. Enter the origin of your user.

If your user's origin is the default identity provider (SAP ID service), choose *sap.ids*.

If your user's origin is a custom platform identity provider, choose the name of the origin or *Other* and enter the name of the origin. The origin of a custom platform identity provider is provided during the configuration of the custom platform identity provider. For more information, see the related link.

7. To assign roles, make sure that the *User & Role Administrator* checkbox is checked.
8. To save your changes, choose *OK*.

## Related Information

[Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#)

[User and Member Management](#)

## 5.1.1.9.2 Remove Security Administrators from Your Subaccount Feature Set A

Running on the cloud management tools feature set A: You can remove security administrators from your subaccount by deleting them.

### Prerequisites

- You can remove users as security administrators: You either created this subaccount, or an authorized user has assigned the `User & Role Administrator` role to your user for it.
- Your user and these users must have at least one of the following member roles (see the related link):
  - They are members of the Cloud Foundry organization (if available) in the subaccount.
  - They are members of the Cloud Foundry spaces that belong to the organization.
  - They are members of the global account that contains your subaccount.

### Context

To remove security administrators from your subaccount, take the following steps.

### Procedure

1. Open the cockpit of SAP Cloud Platform.
  2. Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)). Since your user is the security administrator of your subaccount, you see `Security` in the navigation area.
  3. Choose `Security` `Administrators` .
- You see a list of the security administrators with their respective roles.
4. To remove a security administrator from your subaccount, choose .

### Related Information

[Add Org Members Using the Cockpit \[Feature Set A\]](#) [page 919]

[User and Member Management](#)

## 5.1.1.10 Add Members to Your Subaccount [Feature Set B]

Add members to your subaccount by assigning them a predefined role collection.

### Prerequisites

You must be a subaccount administrator in order to add other subaccount members.

### Context

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

There are 2 predefined role collections that you can use when adding subaccount members:

- Subaccount Administrator
- Subaccount Viewer

For more information on the roles included in each of these role collections, see [Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#).

### Procedure

1. In your subaccount, use the side navigation to go to .
2. Choose the identity provider.
3. Enter the email address of the platform user you want as a subaccount member and choose *Show Assignments*.
4. **Optional:** If the user is not part of the identity provider choose *Add User* in the dialog that appears.
5. **Optional:** Choose *Assign Role Collection*, select the role collection you want to assign to the user and choose *Assign Role Collection* once more in the dialog to confirm.

### Results

The new role collection assigned to the user is displayed in the table. The user is now a subaccount member.

## Related Information

[User and Member Management](#)  
[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.11 Change Subaccount Details

Edit subaccounts using the SAP Cloud Platform 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 Cloud Platform that is generated when the subaccount is created.

You can change to the following subaccount details:

Editable Subaccount Details [Feature Set A]

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	Specify a short descriptive text about the subaccount.
Used for production	Select this option if your subaccount is being used for production purposes. This can help SAP Support to take appropriate action when handling incidents that are related to mission-critical accounts in production systems. This setting does not change your subaccount in any way.

**i Note**

Do not select this option if your subaccount is intended for non-production purposes, such as development, testing, and demos.

Field	Details
Enable beta features	<p>Enable the subaccount to use services and applications which are occasionally made available by SAP for beta usage on SAP Cloud Platform. This option is available to administrators only and is, by default, unselected.</p> <div style="background-color: #f0f0f0; padding: 10px;"> <p><b>⚠ Caution</b></p> <p>You shouldn't use SAP Cloud Platform beta features in subaccounts that belong to productive enterprise accounts. Any use of beta functionality is at the customer's own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.</p> </div> <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p><b>i Note</b></p> <p>Once you have enabled this setting in a subaccount you cannot disable it.</p> </div>

#### Editable Subaccount Details [Feature Set B]

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	Specify a short descriptive text about the subaccount.
Used for production	Select this option if your subaccount is being used for production purposes. This can help your cloud operator to take appropriate action when handling incidents that are related to mission-critical accounts in production systems. This setting does not change your subaccount in any way.
Custom Properties	Assign, change, or remove custom properties from your subaccount. Custom properties help to make organizing and filtering your subaccounts easier.

## 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*.

2. Make your changes and save them.

## Related Information

[Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

[Cloud Management Tools — Feature Set Overview](#)

[Change the Default Database System](#)

### 5.1.1.1.12 Delete Subaccounts

Delete subaccounts using the SAP Cloud Platform cockpit.

#### Prerequisites

You are a global account administrator.

#### Context

You cannot delete a subaccount that contains service instances or active subscriptions to multitenant applications.

#### Procedure

1. Choose the subaccount that you want to delete.
2. Choose *Delete Subaccount* and confirm the operation.

## Related Information

[Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

## 5.1.1.13 Create a Directory (Beta)

Create a directory using the cockpit to organize and manage your subaccounts. For example, you can group subaccounts by project, team, or department.

### Context

#### i Note

This feature is new in Feature Set B so there is no equivalent in Feature Set A. For more information, see [Cloud Management Tools — Feature Set Overview](#).

#### i Note

This is a beta feature.

Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. This means that beta features may be changed by SAP at any time for any reason without notice. Beta features aren't for productive use. Any use of beta functionality is at your own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

For more information on directories, see [Directories \(Beta\) \[Feature Set B\]](#).

### Procedure

1. In your global account, use the left hand-side navigation to go to the [Directories \(beta\)](#) page.
2. Choose [Create Directory](#).
3. In the wizard, enter a display name for your new directory, select any additional features you would like to have in your directory and choose [Next](#).

Based on the additional features you select, you get additional wizard steps. For more information on directory features, see [Directories \(Beta\) \[Feature Set B\]](#).

4. (Optional - Set Custom Properties) Assign custom properties to the directory to make organizing and filtering your directories easier.
5. (Optional - Assign Entitlements) If you selected the [Manage Entitlements](#) feature, you can already assign entitlements and quota to the directory during the creation process. This can also be done at a later stage, after you've created your directory.

To learn more about directory entitlements, see [Configure Entitlements and Quotas for Directories \(Beta\) \[page 779\]](#).

6. (Optional – Manage Directory Authorizations) As a directory administrator, you can already manage authorizations during the creation process if you selected the Manage Authorizations feature. This can also be done at a later stage, after you've created your directory.
7. Finally, review the details of your directory and choose [Create](#) to finalize the process.

## Results

Your directory is created. You can view your directories in 3 different modes, which can be switched using the buttons in the top right-hand corner:

-  Tile View
-  List View
-  Tree View

## 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 custom properties of the directory. You cannot change (enable or disable) the optional features of a directory once it's been created.
- Add subaccounts to the directory - [Add, Move or Remove Subaccounts from Directories \(Beta\) \[Feature Set B\] \[page 774\]](#).
- Delete the directory - you can only delete a directory that contains no subaccounts.

## Related Information

[Cloud Management Tools — Feature Set Overview](#)  
[Custom Properties \[Feature Set B\]](#)

### 5.1.1.1.14 Add, Move or Remove Subaccounts from Directories (Beta) [Feature Set B]

Move a subaccount from a global account to a directory, across directories or from a directory back to the global account using SAP Cloud Platform cockpit.

## Context

### Note

This is a beta feature.

Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. This means that beta features may be changed by SAP at any time for any reason without notice. Beta features aren't for productive use. Any use of beta functionality is at your own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

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.

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.

## Procedure

1. In your global account, navigate to the [Subaccounts](#) page.
2. On the tile of the subaccount you want to move, choose  [Move \(beta\)](#).

This opens a dialog where you can see the subaccount's current parent. This is either the global account (if the subaccount is not part of any directory), or the directory that the subaccount is currently part of.

3. In the dialog, use the dropdown to choose the new parent of the subaccount. You have the following options:
  - Add the subaccount from the global account to a directory.
  - Move the subaccount from one directory to another.
  - Remove the subaccount from a directory by choosing the global account as its parent.
4. Once you've made the selection, choose [Move](#) to confirm.

## Results

Your subaccount has a new parent, which can be seen under  [More Info](#) on the subaccount tile.

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

[Directories \(Beta\) \[Feature Set B\]](#)

[Create a Directory \(Beta\) \[page 773\]](#)

[Create a Subaccount \[Feature Set A\] \[page 762\]](#)

[Navigate to Global Accounts and Subaccounts \[Feature Set B\] \[page 757\]](#)

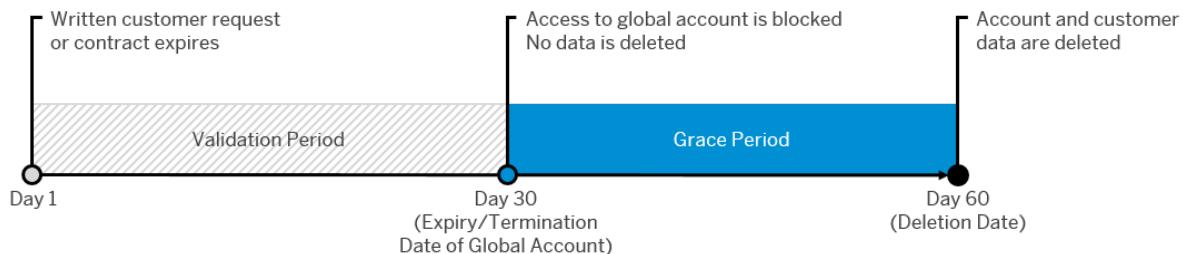
## 5.1.1.15 Global Account Termination

### i Note

For any SAP Cloud Platform instance not running on AWS, Azure, or GCP regions, all references to SAP in this topic are the responsibility of the respective cloud operator.

When the contract of an SAP Cloud Platform customer ends, SAP is legally obligated to delete all the data in the customer's accounts. The data is physically and irreversibly deleted, so that it cannot be restored or recovered by reuse of resources.

The termination process is triggered when a customer contract expires or a customer notifies SAP that they wish to terminate their contract.



1. SAP sends a notification e-mail of the expiring contract, and the termination date on which the account will be closed.
2. A banner is displayed in the cockpit during the 30-day validation period before the global account is closed. During this period, the customer can export their data, or they can open an incident to download their data before the termination date.  
The customer can cancel the termination during this period and renew their contract with SAP.
3. After the termination date, a grace period of 30 days begins.
4. During the grace period:
  - Access is blocked to the account, and to deployed and subscribed applications.
  - No data is deleted, and backups are ongoing.
  - The global account tile is displayed in the *Global Accounts* page of the cockpit with the label *Expired*. Clicking the tile displays the amount of days left in the grace period.The customer can contact SAP to restore their account to a fully active account without data loss.
5. After the end of the grace period, all customer-related data for the account and services is deleted and cannot be restored. The global account tile is removed from the cockpit.

## Related Information

[Getting Support \[page 1161\]](#)

## 5.1.1.2 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 and quotas are managed at the global account level, distributed to subaccounts, and consumed by the subaccounts. When quota is freed at the subaccount level, it becomes available again at the global account level.

### i Note

Only global account administrators can configure entitlements and quotas for subaccounts.

There are two places in the cockpit where you can view and configure entitlements and quotas for subaccounts - at global account level and at subaccount level. Depending on your permissions, you may only have access to one of these pages. You can find more details below:

Entitlements Pages [Feature Set A]

Page in cockpit	Navigation Level	Visible to	Permissions
▶ <a href="#">Entitlements</a> ▶ <a href="#">Subaccount 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>

Entitlements Pages [Feature Set B]

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>

Page in cockpit	Navigation Level	Visible to	Permissions
▶ Entitlements ➔ Service Assignments ➔	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>
Entitlements	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>

You can also assign entitlements to directories, see [Configure Entitlements and Quotas for Directories \(Beta\) \[page 779\]](#).

#### i Note

- [Feature Set B] To subscribe to a multitenant application, you must first assign it as an entitlement to the specific subaccount. If you remove a subaccount's entitlement to a multitenant application, any subscriptions to it from that subaccount will stop working.
- [Feature Set B] 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 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 Quota Plans \[page 927\]](#)
- <https://docs.cloudfoundry.org/adminguide/quota-plans.html> ↗

#### → Tip

Optionally, you can configure entitlements and quotas for supported services that you own or subscribe to from supported non-SAP cloud vendors. These services can be consumed in SAP Cloud Platform 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 \[page 833\]](#).

## Related Information

### Entitlements and Quotas

[Configure Entitlements and Quotas for Subaccounts \[page 781\]](#)

[Configure Entitlements and Quotas for Directories \(Beta\) \[page 779\]](#)

[Create Space Quota Plans \[page 928\]](#)

[Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#)

[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.2.1 Configure Entitlements and Quotas for Directories (Beta)

#### i Note

This feature is new in Feature Set B so there is no equivalent in Feature Set A. For more information, see [Cloud Management Tools — Feature Set Overview](#).

Assign entitlements to directories by adding service plans and distribute the quotas available in your global account to your directories using the SAP Cloud Platform cockpit.

#### Prerequisites

- You are a global account administrator
- Your directory has the manage entitlements feature enabled

#### Context

#### i Note

This is a beta feature.

Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. This means that beta features may be changed by SAP at any time for any reason without notice. Beta features aren't for productive use. Any use of beta functionality is at your own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

#### Procedure

1. Navigate to your global account.
2. Choose  [Entitlements](#)  [Entity Assignments](#)  from the left hand-side navigation.
3. At the top of the page, choose [Show: Directories](#) and then select all the directories for which you would like to configure or display entitlements.

4. Choose [Go](#) to apply the filter.

You get a table for each of the directories you selected, displaying the current entitlement and quota assignments. You can choose [Show subaccount assignments](#) next to the table title to see the entitlement and quota assignments for the subaccounts in a specific directory.

5. Choose [Configure Entitlements](#) to start editing entitlements for a particular directory.

**i Note**

You can only edit entitlements for one directory at a time.

6. You can now edit the entitlements table:

Action	Steps
<b>Add new service plans to the directory</b>	<p>Choose <a href="#">Add Service Plans</a> and from the dialog select the services and the plans from each service that you would like to add to the directory. Choose <a href="#">Add &lt;x&gt; Service Plans</a> in the dialog to confirm.</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 subaccounts. 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>
	<p><b>i Note</b></p> <p>To subscribe a subaccount to a multitenant application, you must first assign the application to the specific subaccount using the process described in <a href="#">Configure Entitlements and Quotas for Subaccounts [page 781]</a>.</p>
	<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 <a href="#">Service Details</a> 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 833]</a>.</p>

**Edit the quota for one or more service plans**

Use + and — to increase or decrease the quota for each service plan.

If you would like to have quota automatically assigned to subaccounts that are created or moved to the directory,

Action	Steps
	<p>follow the same process to set the amount that should be auto-assigned to each subaccount.</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p><b>i Note</b></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></div>

**Delete a service plan and its quota from the directory** Choose  from the *Actions* column.

7. Once you're done, choose [Save](#) to save the changes and exit edit mode for that directory.
8. **Optional:** Repeat steps 5 to 7 to configure entitlements for the other directories selected.

## Related Information

[Entitlements and Quotas](#)

[Cloud Management Tools — Feature Set Overview](#)

[Directories \(Beta\) \[Feature Set B\]](#)

### 5.1.1.2.2 Configure Entitlements and Quotas for Subaccounts

Assign entitlements to subaccounts by adding service plans and distribute the quotas available in your global account to your subaccounts using the SAP Cloud Platform cockpit.

#### Prerequisites

You must be a global account administrator to configure subaccount entitlements.

#### Context

You can distribute entitlements and quotas across subaccounts within a global account from two places in the cockpit:

- The  [Entitlements](#)  [Subaccount Assignments](#) page in the at global account level (only visible to global account administrators)

- The [Entitlements](#) page at subaccount level (visible to all subaccount members, but only editable by global account administrators)

[Feature Set A] To get an overview of all the services and plans available in the global account, you can navigate to [Entitlements](#) [Service Assignments](#) from the left hand-side navigation. There you can see the global usage of each service plan, as well as the detailed assignments of each service across subaccounts, but you will not be able to make any changes on this page.

For more information see [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).

#### Note

- [Feature Set B] 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 entitlement to a multitenant application, any subscriptions to it from that subaccount will stop working.
- [Feature Set B] 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 non-quota-based environments, such as Cloud Foundry, are available by default to all subaccounts, and therefore are not available as entitlements.

## Configure Entitlements and Quotas from Your Global Account

### Procedure

1. Navigate to your global account.
2. Choose [Entitlements](#) [Subaccount Assignments](#) from the left hand-side navigation.

#### Note

[Feature Set B] Choose [Entitlements](#) [Entity Assignments](#) from the left hand-side navigation.

3. At the top of the page, select all the subaccounts for which you would like to configure or display entitlements.

#### Tip

If your global account contains more than 20 subaccounts, choose to open up the value help dialog. There you can filter subaccounts by role, environment and region to make your selection easier and faster. You can only select a maximum of 50 subaccounts at once.

4. Choose [Go](#) to apply the filter.
- You will get a table for each of the subaccounts you selected, displaying the current entitlement and quota assignments.
5. Choose [Configure Entitlements](#) to start editing entitlements for a particular subaccount.

#### Note

You can only edit entitlements for one subaccount at a time.

6. You can now edit the entitlements table:

Action	Steps
<b>Add new service plans to the subaccount</b>	Choose <a href="#">Add Service Plans</a> and from the dialog select the services and the plans from each service that you would like to add to the subaccount.
	<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 <a href="#">Service Details</a> 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 833]</a>.</p>
<b>Edit the quota for one or more service plans</b>	Use <b>+</b> and <b>-</b> to increase or decrease the quota for each service plan.
<b>Delete a service plan and its quota from the subaccount</b>	Choose  from the <a href="#">Actions</a> column.

7. Once you're done, choose [Save](#) to save the changes and exit edit mode for that subaccount.
8. **Optional:** Repeat steps 5 to 7 to configure entitlements for the other subaccounts selected.

## Related Information

[Entitlements and Quotas](#)  
[Cloud Management Tools — Feature Set Overview](#)

## Configure Entitlements and Quotas from Your Subaccount

If you are working in a subaccount and realize you are missing entitlements or quota, you can edit the entitlements 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 entitlements.
2. Choose *Entitlements* from the left hand-side navigation to see the entitlements for your subaccount.
3. Choose *Configure Entitlements*.
4. You can now edit the entitlements table:

Action	Steps
<b>Add new service plans to the subaccount</b>	Choose <i>Add Service Plans</i> and from the dialog select the services and the plans from each service that you would like to add to the subaccount.  <b>→ Tip</b> 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.  For more information, see <a href="#">Managing Resource Providers [page 833]</a> .
<b>Edit the quota for one or more service plans</b>	Use + and – to increase or decrease the quota for each service plan.
<b>Delete a service plan and its quota from the subaccount</b>	Choose  from the <i>Actions</i> column.

5. Once you're happy with the changes, choose *Save* to save and exit edit mode.

## Related Information

[Entitlements and Quotas](#)  
[Cloud Management Tools — Feature Set Overview](#)

### 5.1.1.3 Security Administration: Managing Authentication and Authorization

This section describes the tasks of administrators in the Cloud Foundry environment of SAP Cloud Platform. 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. Now you can manage the authorizations of the business users.

### i Note

Running on the cloud management tools feature set A: Only the administrator who created the current subaccount can use the cockpit of SAP Cloud Platform to add other administrators as members. The added administrators can use all the security administration functions. This enables the administrators to manage authentication and authorization in this subaccount.

## Authentication

Identity providers provide the business users. The default platform identity provider SAP Cloud Platform is SAP ID service. If you use external SAML 2.0 identity providers, you must configure the trust relationship using the cockpit. The respective subaccount must have a trust relationship with the SAML 2.0 identity provider. Using the cockpit, you, as an administrator of the Cloud Foundry environment, establish this trust relationship.

## Authorization

In the Cloud Foundry environment, application developers create and deploy application-based authorization artifacts for business users. Administrators use this information to assign roles, build role collections, and assign these collections to business users or user groups. In this way, they control the users' permissions.

### Setting Up Authorization Artifacts (Administrators)

Step	Task	User Role	Tool
1	<p>Use an existing role or create a new one using role templates</p> <p><a href="#">Add Roles to Role Collections on the Application Level [page 826]</a></p>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
2	<p>Create a role collection and assign roles to it</p> <p><a href="#">Maintain Role Collections [page 828]</a></p>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform

Step	Task	User Role	Tool
3	Assign the role collections to users  <a href="#">Managing Users and Their Authorizations Using the sapcp CLI [page 873]</a> or <a href="#">Assign Role Collections [page 829]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
4	(If you do not use SAP ID Service) Assign the role collections to SAML 2.0 user groups (cloud management tools feature set A regions)  <a href="#">Map Role Collections to User Groups [page 831]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit
5	Assign the role collection to the business users provided by an SAML 2.0 identity provider (cloud management tools feature set A)  <a href="#">Directly Assign Role Collections to Users [page 830]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit

## Related Information

[Trust and Federation with Identity Providers \[page 789\]](#)

[Monitoring and Troubleshooting \[page 1130\]](#)

[SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#)

[SAP ID Service \[page 786\]](#)

### 5.1.1.3.1 SAP ID Service

The default platform identity provider and application identity provider of SAP Cloud Platform is SAP ID service.

## Context

Trust to SAP ID service in your subaccount is pre-configured in both the Neo and the Cloud Foundry environment of SAP Cloud Platform by default, so you can start using it without further configuration. Optionally, you can add additional trust settings or set the default trust to inactive, for example if you prefer to use another SAML 2.0 identity provider. Using the SAP Cloud Platform cockpit you can make changes by

navigating to your respective subaccount and by choosing for Cloud Foundry, and for Neo.

If you want to add new users to a subscribed app, or if you want to add users to a service, such as Web IDE, you can add those users to SAP ID service in your subaccount. See [Add Users to SAP ID Service in the Cloud Foundry Environment \[page 788\]](#)

#### Note

If you want to use a custom IdP, you must establish trust to your custom SAML 2.0 identity provider. We describe a custom trust configuration using the example of SAP Cloud Platform Identity Authentication service.

For more information, see [Trust and Federation with Identity Providers \[page 789\]](#).

## Related Information

[Security \[page 1085\]](#)

[Platform Identity Provider](#)

### 5.1.1.3.1.1 Create SAP User Accounts

If you want to add users to SAP ID service in your subaccount, you must ensure that they have an SAP user account.

## Context

If you register for an SAP Cloud Platform trial account <https://cockpit.hanatrial.ondemand.com>, you automatically get a user in SAP ID service. But if you want to add other users to SAP ID service in your subaccount, you must ensure that they have an SAP user account (for example, an S-user or P-user). This could be the case if you wanted to add new users to a subscribed app in your subaccount.

## Procedure

Choose one of the following ways to create SAP user accounts:

- Use the user management of SAP One Support Launchpad to create S-users.
- Register directly at [SAP ID Service](#).
- Register at [SAP Developer Center](#).

## 5.1.1.3.1.2 Add Users to SAP ID Service in the Cloud Foundry Environment

Before you can assign roles or role collection to a user in SAP ID service, you have to ensure that this user is assigned to SAP ID service.

### Prerequisites

The user you want to add to SAP ID service must have an SAP user account (for example, an S-user or P-user). For more information, see [Create SAP User Accounts \[page 787\]](#).

### Context

When you create your own trial account, your SAP user is automatically created and assigned to SAP ID service. But when you onboard new members to your subscribed app, you must add them to your subaccount and ensure that they are also added to SAP ID service. Then you can assign a role collection to a user in SAP ID service.

### Procedure

1. Go to your subaccount and choose ► [Security](#) ► [Trust Configuration](#) ▾.
2. Choose the trust configuration for SAP ID service.
3. Enter the business user's e-mail address in the [E-Mail Address](#) field, for example [john.doe@example.com](mailto:john.doe@example.com).
4. To see the user's role collection assignments, choose [Show Assignments](#).

#### → Remember

If the user identifier you entered does not have an SAP user account or has never logged on to an application in this subaccount, SAP Cloud Platform cannot automatically verify the user name. To avoid mistakes, you must ensure that the user name is correct and that it exists in SAP ID service.

5. If SAP Cloud Platform displays a message saying that the user is not yet in SAP ID service, choose [Add User](#).

### Related Information

[Platform Identity Provider](#)  
[Assign Role Collections \[page 829\]](#)

## 5.1.1.3.2 Trust and Federation with Identity Providers

When setting up accounts you need to assign users. While we provide you with your first users to get you started, your organization has its own user bases which you want to integrate.

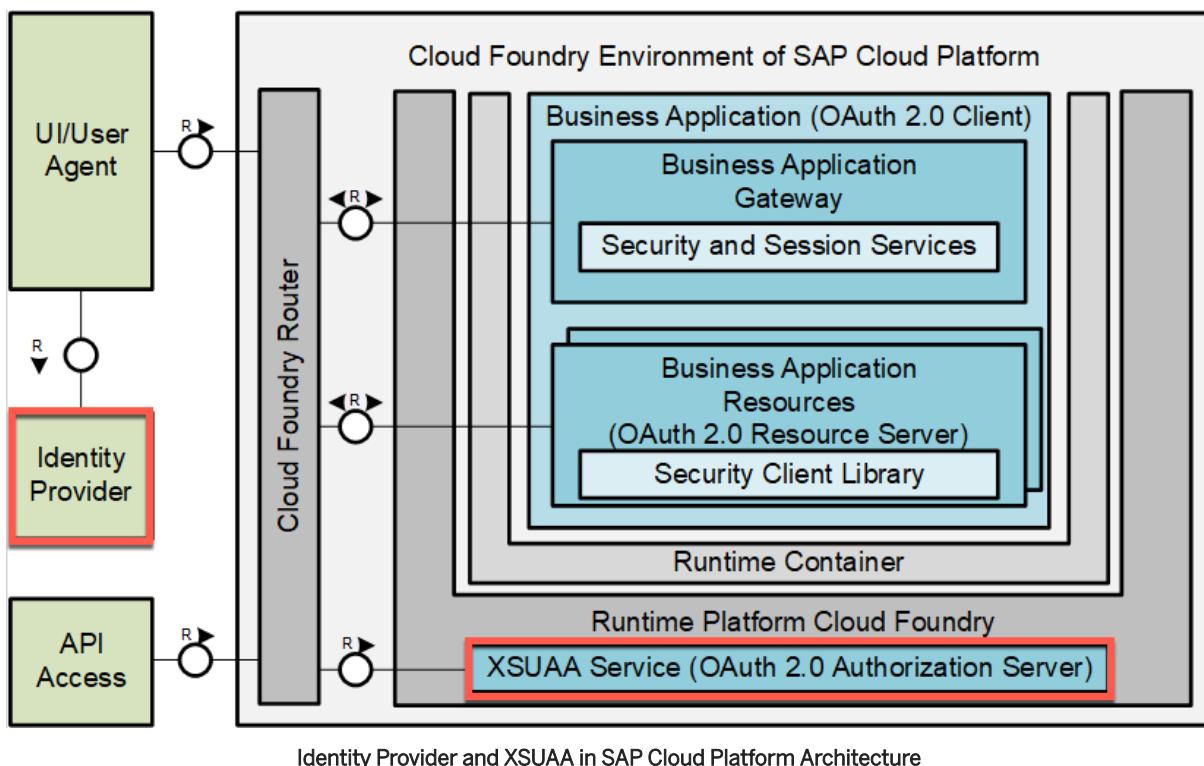
SAP Cloud Platform supports identity federation, a concept of linking and reusing digital identities of a user base across loosely coupled systems. Identity federation frees applications on SAP Cloud Platform from the need to obtain and store the credentials of users to can authenticate them. Instead, 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. To enable communication between SAP Cloud Platform 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 SAP Cloud Platform Identity Authentication service as a hub, especially if your business users are stored in multiple corporate identity providers.

For this scenario, connect Identity Authentication as single custom identity provider to SAP Cloud Platform. Next use Identity Authentication to integrate your corporate identity providers.

For more information, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication](#) and [SAP Cloud Platform Identity Authentication service](#)



Identity Authentication is a multitenancy enabled identity management service for all applications powered by SAP Cloud Platform and optionally on-premise applications. The service provides capabilities for authentication, single sign-on, user provisioning, and on-premise integration as well as self-services like registration or password reset — for both the employees and the partners and customers of your organization.

For administrators, the service offers features for user lifecycle management and reporting capabilities in the administration console.

SAP Cloud Platform has its own Identity Authentication tenant, SAP ID Service. SAP ID Service is the default identity provider of SAP Cloud Platform and where you register to get initial access to SAP Cloud Platform. Trust to SAP ID service is preconfigured by default.

We recommend that you request your own Identity Authentication tenant, but you can also use any other identity provider, which supports the SAML 2.0 protocol. To establish trust with your identity provider, perform one of the following procedures.

- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#)
- [Manually Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service \[page 794\]](#)
- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#)
- [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#)
- [Default Identity Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#)

#### i 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 Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#).

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 [Assign Role Collections \[page 829\]](#).

The identity provider hosts the business users, who belong to user groups. It's efficient to use federation by assigning role collections to one or more user groups. The role collection contains all the authorizations that are necessary for this user group. This method saves time when you add a new business user. Simply add the users to the respective user groups and the new business users automatically get all the authorizations that are included in the role collection.

## Related Information

[Federation Attribute Settings of Any Identity Provider \[page 807\]](#)

## 5.1.1.3.2.1 Establish Trust Between UAA and SAP Cloud Platform Identity Authentication Service with the Identity Provider API

Automatically create a custom OIDC IDP for your XSUAA tenant by establishing a trust between the XSUAA and an IAS tenant.

### Prerequisites

- You must not have any other OIDC IDP besides the default IDP (origin: sap.default) in your subaccount.
- Your XSUAA tenant has to be on the AWS, Azure or Google Cloud Platform landscape.
- You need to create an apiaccess service instance as explained here and use it for all API calls below.
- Obtain an access token to run the API commands as described in [Get API Access \[page 912\]](#).

### Context

In order to create a new or update an existing IAS IDP for your XSUAA tenant, you need to specify a valid IAS tenant.

### Procedure

1. Run the following command, to get a list of all IAS tenants for your XSUAA tenant.

```
curl --location --request GET 'https://api.authentication.<region>.hana.ondemand.com/sap/rest/identity-providers/ias' \
--header 'Authorization: bearer <apiaccess-token>'
```

#### Sample Code

```
curl --location --request GET 'https://api.authentication.<region>.hana.ondemand.com/sap/rest/identity-providers/ias' \
--header 'Authorization: bearer <apiaccess-token>'
```

2. Choose one of the IAS tenant hosts from the returned list and use as it request parameter of the following command to create your new IDP.

```
curl --location --request POST 'https://api.authentication.<region>.hana.ondemand.com/sap/rest/identity-providers' \
--header 'Content-Type: application/json' \
--header 'Authorization: bearer <apiaccess-token>' \
--header 'Content-Type: application/json' \
--data-raw '{
  "type" : "oidc1.0",
```

```
        "config": {
            "iasTenant": {
                "host": "<tenant-host>"
            }
        }
    }'
```

#### ↳ Sample Code

```
curl --location --request POST 'https://api.authentication.stagingaws.hanavlab.ondemand.com/sap/rest/identity-providers' \
--header 'Content-Type: application/json' \
--header 'Authorization: bearer MxOS00MDEyLWEwtbmEtNDNizGQ1...' \
--header 'Content-Type: application/json' \
--data-raw '{
    "type": "oidc1.0",
    "config": {
        "iasTenant": {
            "host": "my-ias-tenant.accounts.ondemand.com"
        }
    }
}'
```

3. Verify the OIDC configuration.

```
curl --location --request GET 'https://api.authentication.<region.hana.ondemand.com/sap/rest/identity-providers' \
--header 'Authorization: bearer <apiaccess-token>'
```

#### ↳ Sample Code

```
curl --location --request GET 'https://api.authentication.eu10.hana.ondemand.com/sap/rest/identity-providers' \
--header 'Authorization: bearer MxOS00MDEyLWEwtbmEtNDNizGQ1...'
```

The answer contains a list of IDPs, which will contain an OIDC IDP with origin `sap.custom`. The config section contains the URL of the specified IAS tenant.

## Results

Your OIDC IDP is now configured and you can use it for example, to obtain a password grant token.

## 5.1.1.3.2.2 Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service

Use your SAP Cloud Platform Identity Authentication service 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 Open ID Connect (OIDC) to establish trust.

### Prerequisites

- You have subaccount administrator permissions or you are a security administrator (cloud management tools feature set A) of this account. For more information, see the related link.
- You've configured a tenant of SAP Cloud Platform Identity Authentication service.  
For more information, see [Configuring Applications](#) in the documentation for Identity Authentication.

### Context

#### ! Restriction

You can only establish trust with a single tenant of Identity Authentication per subaccount using this method.

#### → Recommendation

We recommend that you use SAP Cloud Platform Identity Authentication service as a hub, especially if your business users are stored in multiple corporate identity providers.

For this scenario, connect Identity Authentication as single custom identity provider to SAP Cloud Platform. Next use Identity Authentication to integrate your corporate identity providers.

For more information, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication](#) and [SAP Cloud Platform Identity Authentication service](#)

#### → Tip

We provide APIs so you can perform this procedure programmatically. For more information, see the [Identity Provider Management API](#) on [SAP API Business Hub](#).

### Procedure

1. In the SAP Cloud Platform cockpit, go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)) and choose  .

2. Choose *Establish Trust*.
3. Select an identity provider from the list of available and choose *Establish Trust*.

The identity providers listed are the Identity Authentication tenants associated with your customer account.

## Results

You've configured trust in your tenant of the Identity Authentication service, which is your identity provider.

## Next Steps

If you don't need SAP ID service anymore, set it to inactive.

## Related Information

[Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#)

### 5.1.1.3.2.3 Manually Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service

Use your SAP Cloud Platform Identity Authentication service 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 Identity Authentication tenant and then register your subaccount with the tenant. To complete federation, maintain the federation attributes of the user groups.

## Context

### i 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 UAA and SAP Cloud Platform Identity Authentication Service \[page 793\]](#).

### 5.1.1.3.2.3.1 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 Platform Identity Authentication service. This is where the business users for SAP Cloud Platform are stored.

#### Prerequisites

- You have subaccount administrator permissions or you are a security administrator (cloud management tools feature set A) of this account. For more information, see the related link.
- You have downloaded the SAML 2.0 metadata file from the SAP Cloud Platform Identity Authentication service using the URL `https://<Identity_Authentication_tenant>.accounts.ondemand.com/saml2/metadata`.

#### Example

```
https://my-ias-tenant.accounts.ondemand.com/saml2/metadata
```

For more information, see [Configuring Tenant Settings in the SAP Cloud Platform Identity Authentication service](#).

#### Context

You must establish a trust relationship with an SAML 2.0 identity provider in your subaccount in SAP Cloud Platform. The following procedure describes how you establish trust in the SAP Cloud Platform Identity Authentication service.

#### Procedure

1. In the SAP Cloud Platform cockpit, go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)) and choose .
2. Choose [New Trust Configuration](#).
3. Enter a name and a description that make it clear that the trust configuration refers to the identity provider.

#### i 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 from `https://<Identity_Authentication_tenant>.accounts.ondemand.com/saml2/metadata` 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](#). This will fill the *Subject* and *Issuer* fields with the relevant data from the SAP Cloud Platform Identity Authentication service - your SAML 2.0 identity provider.

The name of the new trust configuration now shows the value <Identity\_Authentication\_tenant>.accounts.ondemand.com. It represents the custom identity provider SAP Cloud Platform Identity Authentication service.

This also fills the fields for the single sign-on URLs and the single logout URLs.
  7. Save your changes.
  8. (Optional, cloud management tools feature set A) If you do not need SAP ID service, set it to inactive.
- You have successfully configured trust in the SAP Cloud Platform Identity Authentication service, which is your SAML 2.0 identity provider.
9. To get the SAML metadata of your subaccount, choose the [SAML Metadata](#) button. You create 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.

## Related Information

[Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#)

### 5.1.1.3.2.3.2 Register SAP Cloud Platform 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 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. The identity provider we use here is the SAP Cloud Platform Identity Authentication service.

## Context

Administrators must configure trust on both sides, in the service provider and in the SAML identity provider. This description covers the side of the identity provider (SAP Cloud Platform Identity Authentication service). The trust configuration on the side of the SAP Cloud Platform Identity Authentication service 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** (case-sensitive). This assertion attribute is required for the assignment of roles. This makes sure that there is a trust relationship between the SAP Cloud Platform Identity Authentication service and the subaccount.

We illustrate the process of configuring trust in the service provider by describing how administrators use the administration console of SAP Cloud Platform Identity Authentication Service to register the subaccount.

To establish trust from a tenant of SAP Cloud Platform Identity Authentication service 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 SAP Cloud Platform Identity Authentication service.

`https://<Identity_Authentication_tenant>.accounts.ondemand.com/admin`

2. To go to the service provider configuration, choose **Applications & Resources** in the menu or the **Applications** tile.

### Note

In the administration console, you find the service providers in **Applications**.

3. To add a new SAML service provider, create a new application by using the **+ Add** button.
4. Choose a name for the application 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 **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.

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.

6. Choose **Default Name ID Format** and select **E-Mail** as a unique attribute. Save your changes.
7. Choose **Assertion Attributes**, use **+Add** to add a multi-value user attribute, and enter **Groups** (case-sensitive) as assertion attribute name for the **Groups** user attribute. Save your changes.

## 5.1.1.3.2.4 Establish Trust and Federation with UAA Using Any SAML Identity Provider

To establish trust, configure the trust configuration of the SAML 2.0 identity provider in your subaccount using the 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.

### Context

#### 5.1.1.3.2.4.1 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 Cloud Platform are stored.

### Prerequisites

- You have subaccount administrator permissions or you are a security administrator (cloud management tools feature set A) of this account. For more information, see the related link.
- 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 must establish a trust relationship with an SAML 2.0 identity provider in your subaccount in SAP Cloud Platform. The following procedure tries to guide you through the trust configuration in your SAML 2.0 identity provider.

### Procedure

1. Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)) and choose  [Security](#)  [Trust Configuration](#)  in the SAP Cloud Platform 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.

**i 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. (Optional, cloud management tools feature set A) If you do not need SAP ID service, set it to inactive.

You have successfully configured trust in your SAML 2.0 identity provider.

9. To get the SAML metadata of your subaccount, choose the *SAML Metadata* button. You create 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.

## Related Information

[Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#)

### 5.1.1.3.2.4.2 Register SAP Cloud Platform 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 assertion 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 assertion attribute name for the **Groups** user attribute. Save your changes.

## Related Information

[Federation Attribute Settings of Any Identity Provider \[page 807\]](#)

### 5.1.1.3.2.5 Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers [Feature Set A]

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

By default, platform users in the Cloud Foundry environment are users in SAP ID service - the default identity provider. To provide access for users in your own identity providers, link them by proxy over a tenant of the Identity Authentication service. If you run a corporate identity provider, connect it to the Identity Authentication service, which has a custom trust configuration with SAP Cloud Platform.

## Prerequisites

- You have an Identity Authentication tenant.

#### ! Restriction

Trial accounts of SAP Cloud Platform do not provide an Identity Authentication tenant. This means that you can't configure a custom platform identity provider in a trial account.

For more information, see the [documentation](#) for SAP Cloud Platform Identity Authentication service.

- You have a subaccount with the Neo environment in addition to your subaccount in the Cloud Foundry environment. You need this subaccount in the Neo environment to access the cockpit. We are using the subaccount in the Neo environment with configured SAML trust to an Identity Authentication tenant, and then establish trust between the Identity Authentication tenant and Cloud Foundry subaccounts.  
For more information, see [Create a Subaccount in the Neo Environment](#) and [Platform Identity Provider](#).
- Make sure that the e-mail addresses of all users are valid in your identity provider since the e-mail address is the unique identifier of users in the Cloud Foundry environment.

## Context

Platform users perform technical development, deployment, and administration tasks. They perform subaccount administration in the 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 SAP ID service.

- Integrate the management of these users with your broader identity management strategy, hosted on your own identity providers. You have control over 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 multi-factor authentication.

For more information, see [Supported Tools and Services When Using Custom Platform Identity Providers \[page 804\]](#).

## Procedure

1. Connect your Identity Authentication tenant to your subaccount in the Neo environment.

For more information, see *Create Trust with the Identity Authentication Tenant* under [Platform Identity Provider](#) in the documentation for SAP Cloud Platform, Neo Environment.

2. In your Identity Authentication tenant, create an application of type [OpenID Connect](#), and configure it. For configuring the Identity Authentication tenant, see [Configure OpenID Connect Application for Resource Owner Password Credentials Flow](#) in the documentation for SAP Cloud Platform Identity Authentication service.

### **i** Note

Use the complete URL of the Identity Authentication tenant starting with `https` as name of the identity provider settings. The OIDC protocol uses the name as the issuer of the Identity Authentication tenant.

Skip creating redirect URIs for now. SAP Support provides the redirect URIs later.

3. In your Identity Authentication application, add a client ID and client secret for the application.

For more information, see [Configure Secrets for API Authentication](#) in the documentation for SAP Cloud Platform Identity Authentication service.

4. Open a ticket with SAP Support (for the component, see [Monitoring and Troubleshooting \[page 1130\]](#)) and provide the following information:
  - Component (`BC-CP-CF-SEC-IAM`)
  - Regions that you want to access using the cockpit
  - **Name** of your OpenID Connect configuration

For example: `https://my-tenant.accounts.ondemand.com`

### **i** Note

Depending on your initial configuration, `https://` may not be included in the name. We recommend using `https://` in the name when initially creating the configuration.

### → Tip

Find the name of your OpenID Connect configuration in your IAS administration console (`https://<tenant ID>.accounts.ondemand.com/admin`) under  [Applications & Resources](#)  [Tenant Settings](#)  [OpenID Connect Configuration](#) under the value **Name**.

- Client ID and client secret of the Identity Authentication application connected to the Cloud Foundry landscape for the OIDC token of your Identity Authentication account

#### Caution

Enter the client secret in the secure area of the ticket.

- Origin that identifies the identity provider during logon using Cloud Foundry command line interface and during the assignment of platform authorizations to users

#### Restriction

The origin has a 1-to-1 relationship with the user base of the Identity Authentication tenant. This origin must be unique for all customers of SAP Cloud Platform. The origin is used in the list of members. It shows in which identity provider the user is stored. You must submit an origin name of 36 7-bit ASCII characters at maximum. Use a name that's easy to remember, for example a name that is derived from your company name and that is similar to the user base. SAP support will get back to you if required.

#### Example

Choose an origin like **acme-prod** that identifies your identity provider if the name of the IAS tenant is a hardly readable identifier like `https://caev6ngwk.accounts.ondemand.com`.

You can use the origin **acme-test** to make it clear that this origin refers to the IAS tenant `https://acme-test.accounts.ondemand.com`.

#### Note

Our operations team needs to make manual adjustments in the SAP Cloud Platform regions to establish trust between the Identity Authentication tenant and Neo regions.

For more information about opening a ticket, see [Getting Support \[page 1161\]](#).

SAP answers with redirect URIs and test instructions to confirm that everything has been set up correctly.

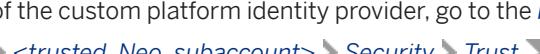
5. In your Identity Authentication application, update your *OpenID Connect Configuration* with the redirect URIs provided by SAP Support.

For more information, see [Configure OpenID Connect Application for Resource Owner Password Credentials Flow](#) in the documentation for SAP Cloud Platform Identity Authentication service.

6. Test the login redirect URI.

With the instructions provided by the support ticket, log on with a user in your custom identity provider.

For more information on the Identity Authentication tenant, see [Configuring Tenant Settings](#).

7. To find the cockpit logon URL of the custom platform identity provider, go to the *Platform Identity Provider* tab of the Neo subaccount in  [Security > Trust](#).
8. To start the logon page of the custom platform identity provider, choose the [Cockpit](#) button.

You can now log on with your user of the custom identity provider.

## Next Steps

- Add platform users.
- Work in the Cloud Foundry command line interface. For more information, see [Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 933\]](#).
- If you want to impose, for example, two-factor authentication to protect your subaccounts, you must configure two-factor authentication in all Identity Authentication applications involved. For more information, see [Configure Risk-Based Authentication](#).
  - In the application of the Identity Authentication tenant for your Neo subaccount (required for cockpit access)
  - In the application of the Identity Authentication tenant for the Cloud Foundry environment

## Related Information

[User and Member Management](#)

### 5.1.1.3.2.5.1 Supported Tools and Services When Using Custom Platform Identity Providers

There are a lot of tools and services to which platform users can log on. The most important ones support custom platform identity providers. This topic describes the current restrictions. It is continually updated as additional tools and services start supporting custom platform identity providers.

Your platform users log on with a custom platform identity provider. When you use such a user, you must take into account that a lot of tools and services are supported, and that there are a number of restrictions.

Supported When Using Platform Identity Providers

Supported	Description
Cockpit	Users logged on using custom platform identity providers can use the full range of the cockpit.
Cloud Foundry command line interface (CF CLI)	Supported. For details see the table with not supported tools and services.
Service Fabrik	You can use the Service Fabrik backing service.
Application Logging and Kibana	Users logged on using custom identity providers can analyze logs written using the application logging service. For more information, see the <a href="#">Application Logging</a> service.

Supported with Restrictions When Using Platform Identity Providers

Supported with Restrictions	Description
SAP Web IDE	SAP Web IDE running with a subaccount of the Cloud Foundry environment of SAP Cloud Platform can only handle users from SAP ID service. For more information, see <a href="#">User Authentication and Authorization</a> .
Cloud Connector	When an administrator connects a Cloud Connector with a subaccount of the Cloud Foundry environment of SAP Cloud Platform, the administrator must use a user from SAP ID service. For more information, see <a href="#">Cloud Connector</a> .
Service dashboards of services that are not listed as supported	If you use services with dashboards, you can't access these dashboards with a user from a custom platform identity provider. Using service dashboards is only possible with the default identity provider (SAP ID service).
Cloud Foundry command line interface (CF CLI)	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> .
<b>! Restriction</b> With v6 of the Cloud Foundry command line interface (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 cockpit. Go to the cockpit under <code>&lt;my_global_account&gt;</code> <code>&lt;my_subaccount&gt;</code> <code>Security</code> <code>Members</code> to set the user's permissions.	
SAP Cloud Platform Rapid Application Development by Mendix	You can't perform operations on CF subaccounts, like deployments with SAP Cloud Platform Rapid Application Development by Mendix when using a custom platform identity provider. For more information, see <a href="#">SAP Cloud Platform Rapid Application Development by Mendix</a> .

## Related Information

[Administration and Operations in the Cloud Foundry Environment \[page 916\]](#)

## 5.1.1.3.2.6 Default Identity Federation with SAP ID Service in the Cloud Foundry Environment

The default identity provider of SAP Cloud Platform is SAP ID Service.

### Prerequisites

- To see the default identity provider in ► [Security](#) > [Trust Configuration](#) ▶, you must have administrator permissions or you are a security administrator of this account. For more information, see the related link.

### SAP ID Service

SAP ID service is the place where you register to get initial access to SAP Cloud Platform. If you are a new user, you can use the self-service registration option of [SAP ID service](#).

Trust to SAP ID service in your subaccount is pre-configured in the Cloud Foundry environment of SAP Cloud Platform by default, so you can start using it without further configuration. Optionally, you can add additional trust settings or set the default trust to inactive (for cloud management tools feature set A, for example if you prefer to use another SAML 2.0 identity provider. Using the SAP Cloud Platform cockpit you can make changes by navigating to your respective subaccount and by choosing ► [Security](#) > [Trust Configuration](#) ▶.

#### i Note

If you do not intend to use SAP ID service, you must establish trust to your custom SAML 2.0 identity provider. We describe a custom trust configuration using the example of SAP Cloud Platform Identity Authentication service.

For more information, see the related link.

### Related Information

[Trust and Federation with Identity Providers \[page 789\]](#)

[Managing Security Administrators in Your Subaccount \[Feature Set A\] \[page 766\]](#)

## 5.1.1.3.2.7 Federation Attribute Settings of Any Identity Provider

This table is supposed to display the attribute settings of the identity provider and the values administrators use to establish trust between the SAML 2.0 identity provider and a new subaccount.

Since there are multiple identity providers you can use, we display the parameters and values of SAP Cloud Platform Identity Authentication service. Use the information in this table as a reference for the configuration of your identity provider.

Settings of SAP Cloud Platform Identity Authentication Service for SAML 2.0 Trust

UI Element	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 UAA service.
<a href="#">Default Name ID Format</a>	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.  Possible settings: <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>	<i>E-Mail</i>  <b>i Note</b> We recommend that you use exactly this name ID format.
<a href="#">Select Assertion Attributes</a>	To define the user authorizations in the UAA service, provide the user groups in the assertion attribute <b>Groups</b> (capitalized). This assertion attribute is required for the assignment of roles in the UAA service.  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.  Some possible settings: <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> You can choose from a number of user attributes and add them.	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 from role collection to user groups has an effect. For more information, see the related link.  <b>⚠ Caution</b> Use exactly this spelling: <b>Groups</b>

### ❖ 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>
        ...
        <Attribute Name="mail">
          <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>
        <Attribute Name="first_name">
          <AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">John</AttributeValue>
        </Attribute>
        <Attribute Name="last_name">
          <AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="xs:string">Doe</AttributeValue>
        </Attribute>
      </AttributeStatement>
    </Assertion>
```

## Related Information

[Map Role Collections to User Groups \[page 831\]](#)

### 5.1.1.3.2.8 Provide Logon Link Help to Identity Provider for Business Users

You have configured trust configurations for multiple identity providers. You want to provide an understandable link on the logon page so that business users know where to log on.

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.

To implement this, configure the following:

1. Hide the default identity provider (`SAP ID service`) from the logon page.  
[Hide Logon Link for Default Identity Provider \[page 809\]](#)
2. Display the custom identity provider at logon time.  
[Display Logon Link for Custom Identity Provider for Business Users \[page 810\]](#)
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 \[page 811\]](#)

### 5.1.1.3.2.8.1 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 Platform Identity Authentication Service, 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 to Orgs and Spaces \[page 917\]](#)) and choose ► *Security* ► *Trust Configuration* in the SAP Cloud Platform 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.

### 5.1.1.3.2.8.2 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 to Orgs and Spaces \[page 917\]](#)) and choose ► *Security* ► *Trust Configuration* in the SAP Cloud Platform 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.

### 5.1.1.3.2.8.3 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 to Orgs and Spaces \[page 917\]](#)) and choose  [Security](#)  [Trust Configuration](#) in the SAP Cloud Platform 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 \[page 810\]](#)

### 5.1.1.3.2.9 Switch Off Automatic Creation of Shadow Users

If you switch off creation of shadow users in the configuration of the trust configuration of custom identity providers, administrators explicitly allow users to log on. Thus, they 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 is no corresponding shadow user, it automatically creates one based on the information received from the identity provider.

Usually, you want your administrators to be fully aware of which users they allow to log on. If you have 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 Cloud Platform cockpit, typically when you assign the first role collection to them.

To switch off creation of shadow users during logon, proceed as follows:

## Procedure

1. In the SAP Cloud Platform cockpit, go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)) and choose .
2. Choose your custom trust configuration.
3. Choose **Edit**.
4. Go to **Create Shadow Users During Logon** and remove the checkmark.

5. Save your changes.

From now on, the User Account and Authentication service does not automatically create shadow users for a certain identity provider during logon.

## Related Information

[Delete Shadow Users for Data Protection and Privacy \[page 1157\]](#)

### 5.1.1.3.3 Working with Role Collections

You can manage role collections by creating new ones from scratch or by copying an existing one and editing it. You can add or remove roles. You can also add or remove users or user groups to the role collections. This is the assignment or unassignment action. You can drill down all the way to the role definition or to the individual role, user, and user group, and make changes there.

## Prerequisites

- Your user has administration rights in this subaccount and or global account. For more information, see [Security Administration: Managing Authentication and Authorization \[page 784\]](#) or [Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#).
- The roles defined by your application developers in the application security descriptor are available in the cockpit. For more information, see [Developing Security Artifacts \[page 251\]](#).
- The users are stored in identity providers that are connected to SAP Cloud Platform.
  - Default identity provider (`SAP_ID_service`). For more information, see [SAP ID Service \[page 786\]](#).
  - Custom identity provider. For more information, see [User Management](#) in SAP Cloud Platform Identity Authentication service.

You can perform the following tasks:

- [Define a Role Collection \[page 814\]](#)
  - [Add Roles to a Role Collection \[page 815\]](#)
  - [Delete Roles from a Role Collection \[page 815\]](#)
- [Assigning Role Collections to Users or User Groups \[page 816\]](#)
  - [Add Users to Role Collections \[page 817\]](#)
  - [Delete Users from Role Collections \[page 818\]](#)
  - [Add User Groups to Role Collections \[page 818\]](#)
  - [Delete User Groups from Role Collections \[page 819\]](#)

## Related Information

[Security Administration: Managing Authentication and Authorization \[page 784\]](#)

[Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#)

### 5.1.1.3.3.1 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
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.

## 5.1.1.3.3.1.1 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
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  (Add a role). Use the dropdown list or the F4 function key under *Role Name* to display the roles that are available. The roles are sorted alphabetically.
7. Choose the role you want to add.
8. Save your changes.
9. Repeat to add more roles.

### Related Information

[Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#)

[Set Up Security Artifacts \[page 273\]](#)

## 5.1.1.3.3.1.2 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
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.

### 5.1.1.3.3.2 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 \[page 784\]](#) or [Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#).
- The roles defined by your application developers in the application security descriptor are available in the cockpit. For more information, see [Developing Security Artifacts \[page 251\]](#).
- The users are stored in identity providers that are connected to SAP Cloud Platform.
  - Default identity provider (SAP ID service). For more information, see [SAP ID Service \[page 786\]](#).
  - Custom identity provider. For more information, see [User Management](#) in SAP Cloud Platform Identity Authentication service.

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 Cloud Platform using trust configurations.

#### i Note

Custom identity providers can provide user groups and users whereas default identity providers can only provide individual users.

- [Add Users to Role Collections \[page 817\]](#)
- [Delete Users from Role Collections \[page 818\]](#)

- Add User Groups to Role Collections [page 818]
- Delete User Groups from Role Collections [page 819]

### 5.1.1.3.3.2.1 Add 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.

##### i Note

The origin key tells you in which identity provider the user is stored. You can find it in ► [Security](#) ► [Trust Configuration](#).

#### Procedure

1. Open the SAP Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
3. Choose ► [Security](#) ► [Role Collections](#).
4. Choose the role collection to which you want to assign users.
5. Go to the [Users](#) section and choose [Edit](#).
6. Choose + (Add a user).
7. Enter the user ID of the user that you want to add. If the user only exists in a connected identity provider, you must choose the origin key of the identity provider and type in the e-mail address.
8. Select the user and save your changes.

You've now added this user to the role collection. The user has all of the authorizations of the role collection.

## 5.1.1.3.3.2.2 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
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.

## 5.1.1.3.3.2.3 Add User Groups to Role Collections

You can assign users groups to a role collection by adding them to the role collection.

### Context

You can assign user groups from custom identity providers to a role collection. After having entered the user group name, choose the identity provider.

### Procedure

1. Open the SAP Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
3. Choose .
4. Choose the role collection to which you want to assign user groups.
5. Go the to *User Groups* section and choose
6. Enter the name of the user group.

### i Note

If you're using SAP Cloud Platform Identity Authentication service, 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.

7. Select the identity provider where the user group is stored.
8. Save your changes.

## Related Information

[User groups in the SAP Cloud Platform Identity Authentication service](#)

### 5.1.1.3.3.2.4 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
3. Choose [Security](#) [Role Collections](#).
4. Choose the role collection from which you want to unassign a user group.
5. Go the to [User Groups](#) section and choose .
6. Confirm the popup and save your changes.

### 5.1.1.3.4 Building Roles and Role Collections for Applications

As an administrator of the Cloud Foundry environment of SAP Cloud Platform, 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 Cloud Platform cockpit, you can assign the role collections to users logging on with SAML 2.0 assertions or SAP ID service.

## Related Information

[Add Roles to Role Collections on the Application Level \[page 826\]](#)

[Maintain Role Collections \[page 828\]](#)

[Assign Role Collections \[page 829\]](#)

[Set Up Security Artifacts \[page 273\]](#)

### 5.1.1.3.4.1 Roles and Role Collections

Usually a role collection consists of one or multiple roles. You can use the 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 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 SAML 2.0 groups if an SAML 2.0 identity provider stores the users. Using the 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 SAML 2.0 user groups containing users logging on with SAML 2.0 assertions.

## Related Information

[Attributes \[page 822\]](#)

[Security Administration: Managing Authentication and Authorization \[page 784\]](#)

### 5.1.1.3.4.2 Create Roles Using Existing Role Templates

Use the cockpit to create a role. You can also assign attributes and add the role to role collections.

#### Context

You can use existing role templates to create new roles. A cockpit wizard helps you to configure new roles.

#### Procedure

1. Open the SAP Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
3. If your role template comes with a subscribed application, choose *Subscriptions* and the subscribed application.
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 **+** (Create a role) action button in the row of the role template that you want to use.

The *Create Role* wizard opens.

7. Overwrite the displayed role name, enter a description, and choose *Next*.
8. Configure the attributes and choose *Next*. 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

[Attributes \[page 822\]](#)

### 5.1.1.3.4.3 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 Cloud Platform 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.

#### i 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 \[page 256\]](#).

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 Cloud Platform, administrators can use the assertion attributes to refine the roles.

#### i Note

If you want to reference attributes from your identity provider, you must know the exact identifier of the assertion attribute. Go to the SAML 2.0 configuration of your identity provider and use the assertion attributes as they are defined there.

There is an exception. The following attributes from the identity provider are automatically mapped:

#### Automatic Mapping of Attributes

Attribute from Identity Provider	Automatically Mapped to Attribute
<i>first_name</i>	<i>given_name</i>
<i>last_name</i>	<i>family_name</i>
<i>mail</i>	<i>email</i>

Attribute from Identity Provider	Automatically Mapped to Attribute
<i>first_name</i>	<i>given_name</i>
<i>last_name</i>	<i>family_name</i>
<i>mail</i>	<i>email</i>

- 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 \[page 823\]](#)

[Specify Attributes in a New Role \[page 825\]](#)

### 5.1.1.3.4.3.1 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 if you want to use the identity provider as the source of the attributes.

#### i Note

In SAP Cloud Platform Identity Authentication Service or any SAML identity provider, you find the attributes in the SAML 2.0 configuration.

## Procedure

1. Open the cockpit for SAP Cloud Platform.

2. Go to your global account (cloud management tools feature set B) or subaccount (cloud management tools feature set A). For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).
3. Choose your space or [Subscriptions](#) (see the related link).
4. Choose the application.
5. Choose .
6. Choose .

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/SAML Attribute
<a href="#">Static</a>	Enter a static value, for example <b>USA</b> to refine the role depending on the country.
<a href="#">Identity Provider (SAML)</a>	Enter an assertion attribute as defined in your identity provider. Check in your identity provider for the exact syntax of the assertion attribute identifier.  In the case of an SAP Cloud Platform Identity Authentication service, you find the attribute identifier in the settings of the SAML assertion attributes of your SAML identity provider under  .
<a href="#">Unrestricted</a>	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.

9. Save your changes.

## Related Information

[Subscribe to Multitenant Applications in the Cloud Foundry Environment Using the Cockpit \[page 32\]](#)

[Cloud Management Tools — Feature Set Overview](#)

[Attributes \[page 822\]](#)

## 5.1.1.3.4.3.2 Specify Attributes in a New Role

As an administrator of the Cloud Foundry environment, 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 if you want to use the identity provider as the source of the attributes.

#### i Note

In SAP Cloud Platform Identity Authentication Service or any SAML identity provider, you find the attributes in the SAML 2.0 configuration.

### Procedure

1. Open the cockpit for SAP Cloud Platform.
2. Go to your global account (China (Shanghai) region) or subaccount. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).
3. Choose your space or *Subscriptions* (see the related link).
4. Choose the application.
5. Choose .
6. To create a new role, choose in the first row.  
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 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/SAML Attribute
<i>Static</i>	Enter a static value, for example <b>USA</b> to refine the role depending on the country.

Source	Value/SAML Attribute
<i>Identity Provider (SAML)</i>	<p>Enter an assertion attribute as defined in your identity provider. Check in your identity provider for the exact syntax of the assertion attribute identifier.</p> <p>In the case of an SAP Cloud Platform Identity Authentication service, you find the attribute identifier in the settings of the SAML assertion attributes of your SAML identity provider under  <a href="#">Applications &amp; Resources</a>  <a href="#">Applications</a>.</p>
<i>Unrestricted</i>	<p><b>Example</b></p> <p>To use the assertion attribute for cost center, you must enter the value <b>cost_center</b>.</p>

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 in the Cloud Foundry Environment Using the Cockpit \[page 32\]](#)  
[Maintain Role Collections \[page 828\]](#)  
[Attributes \[page 822\]](#)

## 5.1.1.3.4.4 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.

#### i 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 Cloud Platform cockpit.
2. Go to your global account and subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
3. Choose *Security* in the navigation pane or go to 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.

#### i 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 (cloud management tools feature set B) or subaccount (cloud management tools feature set A).
  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 \[page 822\]](#)

[Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.3.4.5 Maintain Role Collections

Role collections group together different roles that can be applied to the 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 [Quick Start: Create Role Collections \(with Predefined Roles\) \[page 271\]](#).

As an administrator of the Cloud Foundry environment of SAP Cloud Platform, you can group application roles in role collections. 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 to the users provided by the SAP ID service or by your identity provider or to SAML 2.0 user groups, for example, sales representatives. For more information, see the related link.

### Related Information

[Working with Role Collections \[page 813\]](#)

## 5.1.1.3.4.5.1 Managing SAML 2.0 Identity Provider

If you want to use assertion attributes, set up SAML trust to configure the SAML identity providers (IDP) for runtime of the Cloud Foundry environment. You must perform this step if you want your applications to use SAML assertions as the logon authentication method.

The configuration consists of the following tasks:

- Add a new SAML identity provider (IDP)
- Modify the assertion attributes in the details of an existing SAML identity provider (IDP)
- Manage role collections based on SAML assertions

## 5.1.1.3.5 Assign Role Collections

You have arranged roles in role collections, and now want to assign these role collections to business users.

### Prerequisites

- You are using one or both of the following trust configurations (see the related links):
  - Default trust configuration (SAP ID service)
  - Custom trust configuration (SAP Cloud Platform Identity Authentication service or any SAML 2.0 identity provider)

### Context

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 than in the identity provider or in SAP Cloud Platform. The following options are available:

- Directly assign role collections to users.
- Map role collections to user groups defined in the identity provider. You initially maintain the mapping between user groups and role collections once in SAP Cloud Platform, and maintain group memberships of users in the identity provider.

#### i Note

If you are using the default trust configuration with SAP ID service, you directly assign users to role collections. For more information, see [Default Identity Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#).

However, if you are using a custom trust configuration, for example, with SAP Cloud Platform Identity Authentication service, you can use both options. For more information on configuring the trust between your subaccount and a custom SAML 2.0 identity provider, see [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#).

### Procedure

1. Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).
2. Choose  **Security**  **Trust Configuration** .
3. Choose your active trust configuration.
4. Continue by choosing one of the following options:

Trust Configuration Used	Possible Assignments
Default trust configuration (SAP ID service)	<a href="#">Directly Assign Role Collections to Users [page 830]</a>
Custom trust configuration (SAP Cloud Platform Identity Authentication service or any SAML 2.0 identity provider)	<a href="#">Directly Assign Role Collections to Users [page 830]</a> <a href="#">Map Role Collections to User Groups [page 831]</a>

## Related Information

[Identity Federation \[page 1098\]](#)

[Trust and Federation with Identity Providers \[page 789\]](#)

### 5.1.1.3.5.1 Directly Assign Role Collections to Users

You want to directly assign a role collection to a business user. Running on the cloud management tools feature set A: you can use this option for default and custom trust configurations.

#### Prerequisites

- You have created role collections containing authorizations in the form of roles.

#### Context

If an application developer has defined the role templates accordingly, the roles come with the role templates of the related applications. A role collection can group roles from a set of applications available to your subaccount..

#### Procedure

1. Navigate to the relevant account.

Running on the cloud management tools feature set A: Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)).

Running on the cloud management tools feature set B: Go to your global account.

2. Choose  .

3. Choose the trust configuration for the identity provider of the business user.
4. Choose *Role Collection Assignment*.
5. Enter the business user's name in the *User* field, for example `john.doe@example.com`.

**i Note**

Running on the cloud management tools feature set A: If you are using a custom trust configuration, enter the user name according to the name ID format configured in the identity provider.

If you are using SAP ID Service, enter the e-mail address.

**→ Tip**

If the user identifier you entered has never logged on to an application in this subaccount, SAP Cloud Platform cannot automatically verify the user name. To avoid mistakes, you must check and confirm that the user name is correct.

6. To see the role collections that are currently assigned to this user, choose *Show Assignments*.
7. To assign a role collection, choose *Assign Role Collection*.
8. Choose the name of the role collection you want to assign.
9. Save your changes.

You have assigned a role collection to a business user.

### 5.1.1.3.5.2 Map Role Collections to User Groups

You want to assign a role collection to a user group provided by an SAML 2.0 identity provider that has a custom trust configuration in SAP Cloud Platform. 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 SAML assertion attribute called `Groups`. Each value of the attribute is mapped to a role collection as described in this procedure.

#### Prerequisites

- You have configured your custom SAML 2.0 identity provider and established trust in your subaccount.

**→ Remember**

The name of the trust configuration to is different from `SAP_ID_Service`. The name of a custom trust configuration to SAP Cloud Platform Identity Authentication service could be as follows:

`https://Identity_Authentication_tenant>.accounts.ondemand.com`

- You have configured the identity provider so that it conveys the user's group memberships in the `Groups` assertion attribute.
- You have created role collections containing authorizations in the form of roles.

For more information, see the related links.

## Context

The SAML 2.0 identity provider provides the business users, who can belong to user groups. It is 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. This saves time when you want to add a new business user. Simply add the user to the respective user group or groups, and the business user automatically gets all the authorizations that are included in the role collections.

For this reason, the assignment is a mapping of user groups to role collections.

## Procedure

1. Go to your subaccount (see [Navigate to Orgs and Spaces \[page 917\]](#)) and choose  [Security](#)  [Trust Configuration](#).
2. Choose your custom trust configuration in your subaccount. This identity provider provides the user groups which you want to assign to role collections.
3. Choose [Role Collection Mappings](#).
4. Choose [New Role Collection Mapping](#).
5. Choose the role collection you want to map and enter the name of the user group in the [Value](#) field.

### → Tip

You must know the name of the user group in the identity provider.

### ❖ Example

In the SAP Cloud Platform Identity Authentication service, you find the user groups in the administration console of your SAP Cloud Platform Identity Authentication service tenant under  [Users & Authorizations](#)  [User Groups](#). Open the administration console, for example of SAP Cloud Platform Identity Authentication service using `https://<Identity_Authentication_tenant>.accounts.ondemand.com/admin`.

6. Save your changes.

## Related Information

[Trust and Federation with Identity Providers \[page 789\]](#)

[Manually Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service \[page 794\]](#)

[Federation Attribute Settings of Any Identity Provider \[page 807\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.1.4 Managing Resource Providers

SAP Cloud Platform allows you to connect your global account in the 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.

### i 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 Cloud Platform and consume this service across your subaccounts together with other services offered by SAP.

SAP Cloud Platform 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 Cloud Platform - 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.

#### i 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
	<p><b>Manage Entitlements and Quotas</b></p> <p>View the subaccount entitlements and quotas of services that are consumed from a resource provider.</p> <p>This action opens the  <i>Entitlements</i> <i>Service Assignments</i>  page in the cockpit, and is prefILTERED for the selected resource provider.</p> <div><p><b>→ Tip</b></p><p>Whenever you need to manage assigned entitlements, you can navigate directly to the <i>Service Assignments</i> or <i>Subaccount Assignments</i> pages in the cockpit without going through the <i>Resource Providers</i> page.</p></div>
	<p><b>Edit Resource Provider</b></p> <p>Edit the display name, description, and configuration properties of a resource provider.</p> <div><p><b>i Note</b></p><p>You cannot change the technical name of a resource provider.</p></div>
	<p><b>Delete</b></p> <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 Cloud Platform.</p> <div><p><b>i 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 <i>Subaccount Assignments</i> page.</p></div>

## Related Information

[Configure Entitlements and Quotas for Subaccounts \[page 781\]](#)

### 5.1.1.5 Monitoring Usage and Consumption Costs [Feature Set A]

In a global account that uses the consumption-based commercial model, you can monitor the usage of billed services and your consumption costs.

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

#### i Note

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

## Using the Global Account's Usage Analytics Page

To monitor and track usage in your global account, open the global account in the cockpit and choose [Overview](#) in the navigation area.

The global account's [Overview](#) page provides information about the usage of cloud credits (if your account has a cloud-credit balance) and costs for a global account, and the usage of services in the global account and its subaccounts.

#### i Note

Usage data is processed according to accounting formulas that generate a billing document that aggregates all usage, from all subaccounts, so that it is favorable to customers. There is no unified method to calculate costs based on actual usage.

Costs are displayed according to your contract currency.

#### i Note

The [Overview](#) page also displays usage information for global accounts that use the subscription-based commercial model. Cloud credits and cost information is not relevant for these accounts; therefore, only usage data is displayed.

- When there is a cloud-credit balance for the global account, the cloud-credit usage information is displayed for the current contract period. The total contract duration is split into contract periods (usually of one year each) and the total cloud credits are divided between these periods.

### i Note

If your global account has received a cloud-credit refund at any time during the current contract phase, you may see a difference between your total usage and monthly usage in the chart.

- When there is no cloud-credit balance for the global account, the monthly total costs for the global account are displayed from the contract start date.

### i Note

If your global account has received a refund at any time during the current contract phase, you may see a difference between the displayed monthly costs and your billing document.

- In the resource usage views, use the filters to specify which information to display. The *Period* filter applies only 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.

## Understanding the Views in the Overview Page

View	Description	Action
<b>Global Account Info</b>	Displays general information about the global account, including the number of subaccounts it contains, and the number of regions in which it has these subaccounts.	Click the <i>Subaccounts</i> link to navigate directly to the <i>Subaccounts</i> page.
<b>Cloud Credits Usage</b> Displayed when there is a cloud-credit balance for the global account.	<p>Displays the current balance and monthly usage of cloud credits as a percentage of the cloud credits allocated per contract period.</p> <ul style="list-style-type: none"><li>Cloud-credit information is based on the monthly balance statements for the global account.</li><li>For the time period between the last balance statement and the current date, the chart uses estimated values, which are displayed as striped bars.</li></ul> <p>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.</p> <p>Alerts are displayed when you have used more than 90% and when you reach 100% of your cloud credits.</p>	

View	Description	Action
<b>Global Account Costs</b>  Displayed when there is no cloud-credit balance for the global account.	<p>Displays the total cost per month for usage of services in the global account from the contract start date.</p> <p>All cost information is for all regions in the global account.</p> <p>Actual costs are updated after the monthly statement is generated.</p>	
<b>Global Account Overview</b>	<p>Displays usage and costs of services in the global account. The information is broken down according to service plans. All the regions in which a service plan is available are displayed; however, actual usage is only in the regions of your subaccounts.</p>	<p>Use the <i>View By</i> options to switch the view between service plan usage and costs for the global account.</p> <p>Use the filter to select the environment and service for which you want to view information.</p>
	<p>The charts display information for each month for all service plans in the table or only for the selected row. Provisional data for the current month is displayed as striped bars.</p> <p><b>i Note</b></p> <p>Usage and cost information is updated every 24 hours. Data for the first of the month might not appear until the following day.</p> <p><b>i Note</b></p> <p>Some service plans have differential pricing depending on the region. Usage and costs of a service plan in multiple regions with differential pricing are displayed as separate items.</p>	<p>For the chart display, select a row, and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>
<b>Service Usage</b>	<p>Displays resource usage according to service. Some service plans may display usage according to multiple metrics.</p>	<p>Use the filter to select the service and subaccount for which to display usage values.</p> <p>For the chart display, select a row, and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>
<b>Subaccount Usage</b>	<p>Displays resource usage according to subaccount. Some service plans may display usage according to multiple metrics.</p>	<p>Use the filter to select the subaccount for which to display usage values.</p> <p>For the chart display, select a row and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>

## Using the Subaccount's Overview Page

You can also monitor and track the usage of services in a specific subaccount in the subaccount's *Overview* page. The table in the *Service Usage* view displays the subaccount usage in the current month for service plans of the selected service. The chart displays monthly usage for the selected service plan in the selected period. Use the filters to specify which usage information to display in the table and monthly usage charts. The Period filter is applied to the chart display.

## Exporting Usage and Cost Data Information

You can export the displayed usage and cost data to a spreadsheet document:

- To export usage and cost data from all the views, choose *Export* *All Data*.
- To export only cloud-credit usage data, choose *Export* *Cloud Credits Usage*.

The document with the relevant data is downloaded. The document contains several sheets (tabs). The sheets that are included in the export depend on the commercial model that is used by your global account (and the export option that you selected).

Sheet Name	Description	Commercial Model	
		Subscription-Based	Consumption-Based
Global Account Info	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 contract period, is also shown.	Yes	Yes
Global Account Costs	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
Subaccount Costs by Service	Allows you to view monthly usage data and costs of all billable services consumed by plan and subaccount.  All subaccount calculations are estimated and proportionate to the total global account usage:  [Subaccount usage / Global account usage x Rate plan per SKU]	No	Yes

Sheet Name	Description	Commercial Model	
		Subscription-Based	Consumption-Based
Actual Usage	<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

### • Example

A global account, which uses the consumption-based commercial model, has the following usage data in a given month, where SAP HANA and Application Runtime are billable services and Bandwidth is a non-billable service:

Subaccount 1	Space A	1x SAP HANA 256 GB, 200 MB of Application Runtime, and 300 MB of Bandwidth
	Space B	200 MB of Application Runtime and 300 MB of Bandwidth
Subaccount 2	Space C	1x SAP HANA 512 GB, 600 MB of Application Runtime, and 300 MB of Bandwidth

Based on these usage measurements, you would see the following data in the spreadsheet. Each listed item represents one row in the spreadsheet. Cost prices shown are for illustration purposes only and do not reflect the actual rates for the mentioned services.

Sheet	Sample Usage and Cost Data Displayed in Spreadsheet
Global Account Costs	<ul style="list-style-type: none"> <li>1 instance of SAP HANA 256 = EUR 1024</li> <li>1 instance of SAP HANA 512 = EUR 2048</li> <li>1 GB of Application Runtime = EUR 100 + estimated cost distribution to subaccounts</li> </ul>
Subaccount Costs by Service	<ul style="list-style-type: none"> <li>Subaccount 1: 1 instance of SAP HANA 256 = EUR 1024</li> <li>Subaccount 2: 1 instance of SAP HANA 512 = EUR 2048</li> <li>Subaccount 1: 400 MB of Application Runtime = EUR 40 (400 MB / 1 GB x EUR 100)</li> <li>Subaccount 2: 600 MB of Application Runtime = EUR 60 (600 MB / 1 GB x EUR 100)</li> </ul>

Sheet	Sample Usage and Cost Data Displayed in Spreadsheet
Actual Usage	<ul style="list-style-type: none"> <li>Subaccount 1   Space A: 1 instance of SAP HANA 256</li> <li>Subaccount 2   Space C: 1 instance of SAP HANA 512</li> <li>Subaccount 1   Space A: 200 MB of Application Runtime</li> <li>Subaccount 1   Space B: 200 MB of Application Runtime</li> <li>Subaccount 2   Space C: 600 MB of Application Runtime</li> <li>Subaccount 1   Space A: 300 MB of Bandwidth</li> <li>Subaccount 1   Space B: 300 MB of Bandwidth</li> <li>Subaccount 2   Space C: 300 MB of Bandwidth</li> </ul>

## Related Information

[Cloud Management Tools — Feature Set Overview](#)  
[What Is the Consumption-Based Commercial Model?](#)  
[View Subaccount Usage Analytics \[page 847\]](#)

### 5.1.1.6 Monitoring Usage and Consumption Costs [Feature Set B]

In a global account that uses the consumption-based commercial model, you can monitor the usage of billed services and your consumption costs.

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

#### i Note

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

## Using the Global Account's Usage Analytics Page

To monitor and track usage in your global account, open the global account in the cockpit and choose [Usage Analytics](#) in the navigation area.

The global account's [Usage Analytics](#) page provides information about the usage of cloud credits (if your account has a cloud credits balance) and costs for a global account, and the usage of services in the global account and its directories and subaccounts.

### i Note

Usage data is processed according to accounting formulas that generate a billing document that aggregates all usage, from all subaccounts, so that it is favorable to customers. There is no unified method to calculate costs based on actual usage.

Costs are displayed according to your contract currency.

### i Note

The [Usage Analytics](#) page also displays usage information for global accounts that use the subscription-based commercial model. Cloud credits and cost information are not relevant for these accounts; therefore, only usage data is displayed.

- When there is a cloud-credit balance for the global account, the cloud-credit usage information is displayed for the current contract period. The total contract duration is split into contract periods (usually of one year each) and the total cloud credits are divided between these periods.

### i Note

If your global account has received a refund of cloud credits at any time during the current contract phase, you may see a difference between your total usage and monthly usage in the chart.

- When there is no cloud-credit balance for the global account, the monthly total costs for the global account are displayed from the contract start date.

### i Note

If your global account has received a refund at any time during the current contract phase, you may see a difference between the displayed monthly costs and your billing document.

- In the resource usage views, use the filters to specify which information to display. The [Period](#) filter applies only 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.

## Understanding the Views in the Usage Analytics Page

View	Description	Action
Global Account Info	Displays general information about the global account, including the number of directories and subaccounts it contains, and the number of regions in which it has its subaccounts.	Click the <a href="#">Directories</a> link to navigate directly to the <a href="#">Directories</a> page. Click the <a href="#">Subaccounts</a> link to navigate directly to the <a href="#">Subaccounts</a> page.

View	Description	Action
<b>Cloud Credits Usage</b>	<p>Displays the current balance and monthly usage of cloud credits as a percentage of the cloud credits allocated per contract period.</p> <p>Displayed when there is a cloud-credit balance for the global account.</p> <ul style="list-style-type: none"> <li>Cloud-credit information is based on the monthly balance statements for the global account.</li> <li>For the time period between the last balance statement and the current date, the chart uses estimated values, which are displayed as striped bars.</li> </ul> <p>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.</p> <p>Alerts are displayed when you have used more than 90% and when you reach 100% of your cloud credits.</p>	
<b>Global Account Costs</b>	<p>Displays the total cost per month for usage of services in the global account from the contract start date.</p> <p>Displayed when there is no cloud-credit balance for the global account.</p> <p>All cost information is for all regions in the global account.</p> <p>Actual costs are updated after the monthly statement is generated.</p>	

View	Description	Action
Global Account Overview	<p>Displays usage and costs of services in the global account. The information is broken down according to service plans. All the regions in which a service plan is available are displayed; however, actual usage is only in the regions of your subaccounts.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <b>i Note</b>            Usage and cost information is updated every 24 hours. Data for the first of the month might not appear until the following day.         </div> <p>The charts display information for each month for all service plans in the table or only for the selected row. Provisional data for the current month is displayed as striped bars.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <b>i Note</b>            Some service plans have differential pricing depending on the region. Usage and costs of a service plan in multiple regions with differential pricing are displayed as separate items.         </div>	<p>Use the <a href="#">View By</a> options to switch the view between service plan usage and costs for the global account.</p> <p>Use the filter to select the environment and service for which you want to view information.</p> <p>For the chart display, select a row, and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>
Service Usage	<p>Displays resource usage according to service. Some service plans may display usage according to multiple metrics.</p>	<p>Use the filter to select the service and directory or subaccount for which to display usage values.</p> <p>For the chart display, select a row, and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>
Directory/Subaccount Usage	<p>Displays resource usage according to directory or subaccount. Some service plans may display usage according to multiple metrics.</p>	<p>Use the filter to select the directory or subaccount for which to display usage values.</p> <p>For the chart display, select a row and filter by period. If no row is selected, the chart displays information for all the service plans in the table.</p>

## Exporting Usage and Cost Data Information

You can export the displayed usage and cost data to a spreadsheet document:

- To export usage and cost data from all the views, choose [Export > All Data](#).
- To export only cloud-credit usage data, choose [Export > Cloud Credits Usage](#).

The document with the relevant data is downloaded. The document contains several sheets (tabs). The sheets that are included in the export depend on the commercial model that is used by your global account (and the export option that you selected).

Sheet Name	Description	Commercial Model	
		Subscription-Based	Consumption-Based
Global Account Info	<p>Provides general information about your global account. If there is a cloud-credit balance for the global account, then it also shows the cloud-credit usage, per month as a percentage of your total cloud credits for the current contract period.</p>	Yes	Yes
Global Account Costs	<p>Allows you to view total monthly usage data and costs for all billable services and plans consumed at the level of your global account.</p> <p>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.</p>	No	Yes
Subaccount Costs by Service	<p>Allows you to view monthly usage data and costs of all billable services consumed by plan and subaccount. You can also determine the costs at the level of your directories.</p> <p>All subaccount calculations are estimated and proportionate to the total global account usage:</p> $[\text{Subaccount usage} / \text{Global account usage} \times \text{Rate plan per SKU}]$	No	Yes
Actual Usage	<p>Allows to you to view the actual monthly usage data of all consumed services by plan, directory, subaccount, and space.</p> <p>Actual or raw usage data is different than 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

## • Example

A global account, which uses the consumption-based commercial model, has the following usage data in a given month, where SAP HANA and Application Runtime are billable services and Bandwidth is a non-billable service:

Subaccount 1	Space A	1x SAP HANA 256 GB, 200 MB of Application Runtime, and 300 MB of Bandwidth
	Space B	200 MB of Application Runtime and 300 MB of Bandwidth
Subaccount 2	Space C	1x SAP HANA 512 GB, 600 MB of Application Runtime, and 300 MB of Bandwidth

Based on these usage measurements, you would see the following data in the spreadsheet. Each listed item represents one row in the spreadsheet. Cost prices shown are for illustration purposes only and do not reflect the actual rates for the mentioned services.

Sheet	Sample Usage and Cost Data Displayed in Spreadsheet
Global Account Costs	<ul style="list-style-type: none"><li>1 instance of SAP HANA 256 = EUR 1024</li><li>1 instance of SAP HANA 512 = EUR 2048</li><li>1 GB of Application Runtime = EUR 100 + estimated cost distribution to subaccounts</li></ul>
Subaccount Costs by Service	<ul style="list-style-type: none"><li>Subaccount 1: 1 instance of SAP HANA 256 = EUR 1024</li><li>Subaccount 2: 1 instance of SAP HANA 512 = EUR 2048</li><li>Subaccount 1: 400 MB of Application Runtime = EUR 40 (400 MB / 1 GB x EUR 100)</li><li>Subaccount 2: 600 MB of Application Runtime = EUR 60 (600 MB / 1 GB x EUR 100)</li></ul>
Actual Usage	<ul style="list-style-type: none"><li>Subaccount 1   Space A: 1 instance of SAP HANA 256</li><li>Subaccount 2   Space C: 1 instance of SAP HANA 512</li><li>Subaccount 1   Space A: 200 MB of Application Runtime</li><li>Subaccount 1   Space B: 200 MB of Application Runtime</li><li>Subaccount 2   Space C: 600 MB of Application Runtime</li><li>Subaccount 1   Space A: 300 MB of Bandwidth</li><li>Subaccount 1   Space B: 300 MB of Bandwidth</li><li>Subaccount 2   Space C: 300 MB of Bandwidth</li></ul>

## Related Information

[Cloud Management Tools — Feature Set Overview](#)  
[What Is the Consumption-Based Commercial Model?](#)

## 5.1.1.7 View Subaccount Usage Analytics

You can explore, compare, and analyze all your 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 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 <a href="#">Services</a> view.

#### i 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.

#### → Tip

- [Feature Set B] If you use directories to group your subaccounts, you can go to your global account's [Usage Analytics](#) page to view usage information by directory. See [Monitoring Usage and Consumption Costs \[Feature Set B\]](#) [page 841].
- [Feature Set A] 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 [Overview](#) and subaccount's [Overview](#) pages. The global account's [Overview](#) page also provides information about cloud-credit usage. See [Monitoring Usage and Consumption Costs \[Feature Set A\]](#) [page 836].

## 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 \[page 916\]](#)

[Cloud Management Tools — Feature Set Overview](#)

[Monitoring Usage and Consumption Costs \[Feature Set A\] \[page 836\]](#)

[Monitoring Usage and Consumption Costs \[Feature Set B\] \[page 841\]](#)

## 5.1.1.8 Configure Legal Information

Administrators can define legal links per enterprise global account in the SAP Cloud Platform 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 to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

[Cloud Management Tools — Feature Set Overview](#)

## 5.1.2 Account Administration Using the CLI for SAP Cloud Platform (sapcp CLI) [Feature Set B]

The CLI for SAP Cloud Platform (sapcp CLI) is 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.

### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

### Learn How to Work With the sapcp CLI

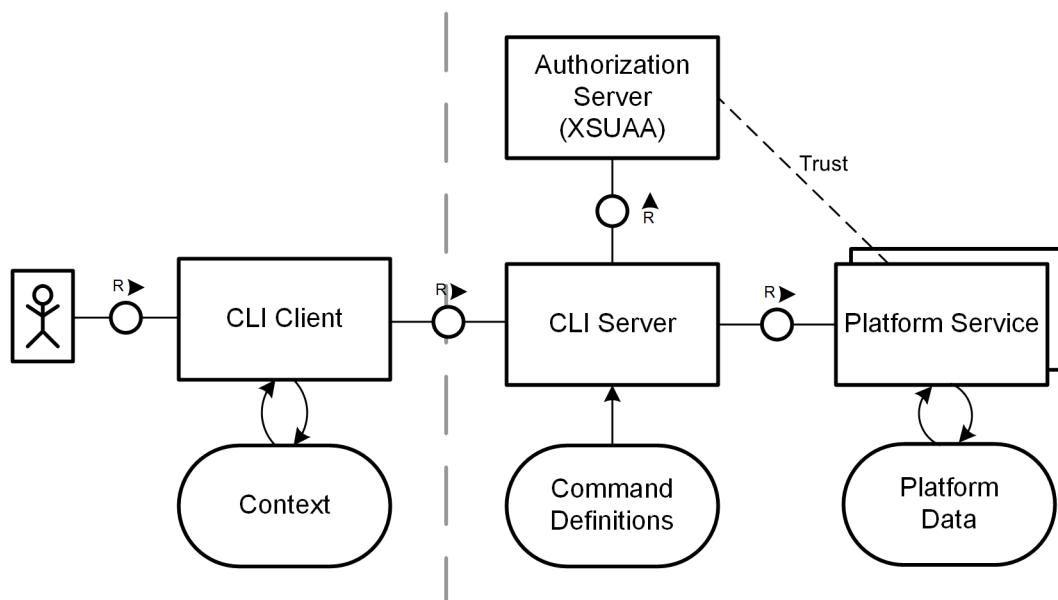
Learn how to use the client:

- [Download and Start Using the Client \[page 851\]](#)
- [Set the Default Command Context \[page 864\]](#)

Learn about frequent administrative tasks you can perform using the sapcp CLI:

- [Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)
- [Setting Entitlements Using the sapcp CLI \[page 870\]](#)
- [Working with Environments Using the sapcp CLI \[page 871\]](#)
- [Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)
- [Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)
- [Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

### How the CLI for SAP Cloud Platform Works



You download the CLI client to your local desktop and access it through the shell of your operating system. The CLI 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.

## A Simple Workflow to Set up a Global Account with the CLI for SAP Cloud Platform (sapcp CLI) and the Cloud Foundry CLI (cf CLI)

The CLI for SAP Cloud Platform (sapcp CLI) is an alternative to the cockpit that lets you carry out all account administration tasks on global account and subaccount level via the command line. If you go down the hierarchy starting with the global account, the last task in the sapcp CLI is to create environment instances (for example, a Cloud Foundry org). From here on, you need the cf CLI if you want to stay in the command line.

For information about setting up a global account with the CLIs, see:

- For trial accounts: [Setting Up a Trial Account via the Command Line \[Feature Set B\] \[page 17\]](#)
- For enterprise accounts: [Setting Up a Global Account via the Command Line \[Feature Set B\] \[page 25\]](#)

## Related Information

[Download and Start Using the Client \[page 851\]](#)

[Command Syntax \[page 855\]](#)

[Login \[page 861\]](#)

[Set the Default Command Context \[page 864\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

[Cloud Management Tools — Feature Set Overview](#)

### 5.1.2.1 Download and Start Using the Client

To use the CLI for SAP Cloud Platform (sapcp CLI), you need to download the client first.

## Context

The client is available for 64-bit versions of the following operating systems:

- Microsoft Windows
- Apple macOS
- Linux

## Procedure

1. Download the appropriate client for your operating system at <https://tools.hana.ondemand.com/#cloud-sapcpcli> or use the direct links to the latest version below:
  - o [Latest version of CLI client for Microsoft Windows](#)
  - o [Latest version of CLI client for Apple macOS](#)
  - o [Latest version of CLI client for Linux](#)

### i Note

You can also find older versions of the client at: <https://tools.hana.ondemand.com/#cloud-sapcpcli>. In exceptional cases, when your client is too new for the configured server, an error message tells you which version you need and where to find it.

2. Extract the client file (for example: `sapcp.exe`) to your local system, put it in your PATH, and open the command line in the target folder.
3. Run `sapcp`. Note that you need read and write permissions in the target folder to run this executable.

### i Note

If you are using macOS, opening `sapc CLI` may be blocked because it is from "an unidentified developer". Please refer to the macOS documentation to learn how to bypass this.

You get a short welcome message with some useful information, such as the version of your client, how to display help, and how to log in.

The CLI creates a configuration file (`config.json`) in your user data directory.

4. Log in to SAP Cloud Platform. Login is always in a **global account**. See [Login \[page 861\]](#). Note that you need to provide the **subdomain of the global account** to log in. You can find this in the global account view in the cockpit. If you have more than one global account, see the [Switch Global Account](#) dialog.
5. Once you're logged in, familiarize yourself with the `sapcp` CLI, for example with [General Commands and Options in the sapcp CLI \[page 854\]](#), [Command Syntax \[page 855\]](#), or simply by trying out a few commands, such as the following:

```
sapcp list accounts/subaccount  
  
sapcp list security/user  
  
sapcp get security/user "name@example.com"  
  
sapcp get accounts/global-account"
```

6. If you're going to work in a subaccount of this global account, consider setting the default context to this subaccount using `sapcp target --subaccount <ID>`. See [Set the Default Command Context \[page 864\]](#). Tip: Use `sapcp list accounts/subaccount` to see the subaccounts of the global account.
7. To find out the current context, use `sapcp`.

## Related Information

[Set the Default Command Context \[page 864\]](#)

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

[General Commands and Options in the sapcp CLI \[page 854\]](#)

### 5.1.2.1.1 Get Updates

We recommend updating the CLI client regularly to ensure that you can always use all new features.

#### Context

To find out the version of the CLI client you are using, run the command `sapcp --info` or simply `sapcp`.

#### Procedure

1. Get the latest version using the download links at: <https://tools.hana.ondemand.com/#cloud-cpcli>.
2. Extract the client file (for example: `sapcp.exe`) and replace the old file with this new one.
3. Work with the sapcp CLI as usual and enjoy the new features.

## Related Information

[Login \[page 861\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

## 5.1.2.2 General Commands and Options in the sapcp CLI

Learn how to work with the CLI for SAP Cloud Platform (sapcp CLI). For example, how to log in, get help, and set a default context for commands.

### General Commands

```
sapcp login
```

```
sapcp logout
```

```
sapcp target
```

### General Options for Each Command

The following options are available for each command. They need to be typed right after the base call `sapcp`, and they can be combined (for example, `sapcp --verbose --help list accounts/subaccount`). The `--help` option also works at the end of a command call.

Options	
<code>--config</code>	Specifies the location of the configuration file. See <a href="#">Specify the Location of the Configuration File [page 866]</a> .
<code>--info</code>	Displays version and current context. Note that this option only works on its own ( <code>sapcp --info</code> ) and cannot be added to other command calls. You can also just use <code>sapcp</code> to display the info. See <a href="#">View Version and Current Context [page 860]</a> .
<code>--help</code>	Displays help. See <a href="#">Get Help [page 859]</a> .
<code>--verbose</code>	Prints tracing information for support. See <a href="#">Troubleshooting and Support [page 878]</a> .

#### [Command Syntax \[page 855\]](#)

Each command consists of the base call `sapcp` followed by a verb (the action), a combination of group and object, and parameters.

#### [Get Help \[page 859\]](#)

Get help in the sapcp CLI with the `--help` option.

#### [View Version and Current Context \[page 860\]](#)

To find out the current context you're working in, run the command `sapcp --info` or simply `sapcp`.

#### [Login \[page 861\]](#)

Login with the CLI for SAP Cloud Platform (sapcp CLI) is on global account level.

#### [Logout \[page 863\]](#)

Logging out of the configured server removes all user-specific data from the configuration file.

#### [Set the Default Command Context \[page 864\]](#)

Change the default context for all command calls to the global account, a directory, or a subaccount by using the `sapcp target` command.

#### [Specify the Location of the Configuration File \[page 866\]](#)

You can change the location of the configuration file by using the `--config` option.

## Related Information

[Commands in the sapcp CLI \[page 867\]](#)

### 5.1.2.2.1 Command Syntax

Each command consists of the base call `sapcp` followed by a verb (the action), a combination of group and object, and parameters.

The `sapcp` CLI uses the following syntax:

```
sapcp [OPTIONS] ACTION GROUP/OBJECT [PARAMS]
```

The commands are ordered in groups. Words in caps are placeholders, and brackets [ ] denote optionality. Here's one example with the help option and no parameters before we outline the entire syntax:

#### Sample Code

```
sapcp --help list accounts/subaccount
```

- `sapcp` is the base call to start each command
- `OPTIONS` can be added to each command, for example to get help `--help` or execute a command in verbose mode `--verbose`. Note that the `--help` option can also be placed at the end of a command.
- `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 `sapcp --help`.
- 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.
- `PARAMETERS` are passed with most commands. 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 `sapcp 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).

## i 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 in the Command Line \[page 857\]](#).

## A Few Commands to Get Started

Here are a few commands for you to try out once you're logged in ([Login \[page 861\]](#)):

```
sapcp list accounts/subaccount  
sapcp list security/user  
sapcp get security/user "name@example.com"  
sapcp list account/subscription
```

## Example

In this example, we assign the **Global Account Administrator** role collection to user `name@example.com` and try out some options.

Using the `--help` option to display command-specific help to learn how to use the command:

```
sapcp --help assign security/role-collection  
sapcp assign security/role-collection --help
```

Command with positional parameter and one mandatory parameter:

```
sapcp assign security/role-collection "Global Account Administrator" --to-user  
"name@example.com"
```

Command with positional parameter, mandatory parameter, and optional parameter:

```
sapcp assign security/role-collection "Global Account Administrator" --to-user  
"name@example.com" --of-idp "my-idp"
```

The same command in verbose mode:

```
sapcp --verbose assign security/role-collection "Global Account Administrator" --  
to-user "name@example.com" --of-idp "my-idp"
```

**Parent topic:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Get Help \[page 859\]](#)

[View Version and Current Context \[page 860\]](#)

[Login \[page 861\]](#)

[Logout \[page 863\]](#)

[Set the Default Command Context \[page 864\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

[Set the Default Command Context \[page 864\]](#)

### 5.1.2.2.1.1 Passing JSON Parameters in the Command Line

Depending on the shell you use, the escaping rules are different. Find examples of correct escaping and quotes when passing JSON objects in the command line using different shells.

The table below gives an overview of the escaping rules in the three most commonly used shells.

Shell	Correct Escaping and Quotes
Bash	Use --parameter "VALUE" and escape quotes with \" within VALUE
	Use --parameter 'VALUE' and do not escape quotes within VALUE
Windows Command Prompt	Use --param "VALUE" and escape quotes with \" within VALUE
Windows Power Shell	Use --param 'VALUE' 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 below are based on the following JSON:

```
{  
[ {  
    "key": " Key 1",  
    "value": "Value 1"},  
{  
    "key": "Key 2",  
    "value": "Value 2"  
}]  
}
```

## Bash (Linux, macOS)

Option 1

```
--parameter '[{"key": "Key 1", "value": "Value 1"}, {"key": "Key 2", "value": "Value 2"}]'
```

For example, when creating a subaccount (`sapcp create accounts/subaccount`) with custom properties `Landscape = Dev` and `Department = HR`, use the following syntax:

↳ Sample Code

```
--custom-properties '[{"key": "Landscape", "value": "Dev"}, {"key": "Department", "value": "HR"}]'
```

Option 2

```
--parameter "[{\"key\": \"Key 1\", \"value\": \"Value 1\"}, {\"key\": \"Key 2\", \"value\": \"Value 2\"}]"
```

For example, when creating a subaccount (`sapcp create accounts/subaccount`) with custom properties `Landscape = Dev` and `Department = HR`, use the following syntax:

↳ Sample Code

```
--custom-properties "[{\"key\": \"Landscape\", \"value\": \"Dev\"}, {\"key\": \"Department\", \"value\": \"HR\"}]"
```

## Windows Command Prompt

```
--parameter "[{\"key\": \"Key 1\", \"value\": \"Value 1\"}, {\"key\": \"Key 2\", \"value\": \"Value 2\"}]"
```

For example, when creating a subaccount (`sapcp create accounts/subaccount`) with custom properties `Landscape = Dev` and `Department = HR`, use the following syntax:

↳ Sample Code

```
--custom-properties "[{\"key\": \"Landscape\", \"value\": \"Dev\"}, {\"key\": \"Department\", \"value\": \"HR\"}]"
```

## Windows Power Shell

```
--parameter '[{"key": \"Key 1\", \"value\": \"Value 1\"}, {\"key\": \"Key 2\", \"value\": \"Value 2\"}]'
```

For example, when creating a subaccount (`sapcp create accounts/subaccount`) with custom properties `Landscape = Dev` and `Department = HR`, use the following syntax:

#### ↳ Sample Code

```
--custom-properties --custom-properties '[{"key": "Landscape", "value": "Dev"}, {"key": "Department", "value": "HR"}, {"key": "Flagged", "value": "\\"}]'
```

## Related Information

[Command Syntax \[page 855\]](#)

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

[Working with Environments Using the sapcp CLI \[page 871\]](#)

## 5.1.2.2.2 Get Help

Get help in the sapcp CLI with the `--help` option.

- To display general help, use `sapcp --help`. This lists all commands, ordered in group/object combinations, with the possible actions.
- To display help for a specific command, type `--help` at the beginning or end of the the respective command. This displays information about the usage, its parameters, a description, helpful tips, and examples.

## Example

#### ↳ Sample Code

```
sapcp --help list accounts/subaccount
```

#### ↳ Sample Code

```
sapcp list accounts/subaccount --help
```

```
>sapcp --help list accounts/subaccount
Usage: sapcp [OPTIONS] list accounts/subaccount
List all the subaccounts in a global account, including the subaccounts in directories.

Example:
  sapcp list accounts/subaccount

Tips:
  You must be assigned to the global account admin or viewer role.
  To view all directories and subaccounts in a global account, use 'sapcp get accounts/global-account --show-hierarchy'.
  To view the subaccounts in a specific directory, use 'sapcp get accounts/directory'.

See also:
  sapcp get accounts/subaccount
  sapcp get accounts/global-account
  sapcp get accounts/directory

OK
```

**Parent topic:** General Commands and Options in the sapcp CLI [page 854]

## Related Information

[Command Syntax \[page 855\]](#)

[View Version and Current Context \[page 860\]](#)

[Login \[page 861\]](#)

[Logout \[page 863\]](#)

[Set the Default Command Context \[page 864\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

### 5.1.2.2.3 View Version and Current Context

To find out the current context you're working in, run the command `sapcp --info` or simply `sapcp`.

#### Procedure

Use `sapcp --info` or just `sapcp`.

The client displays its own version, usage information, the CLI server URL, the global account, directory, and subaccount you're working in and their hierarchy, as well as the location of the configuration file.

**Task overview:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[Login \[page 861\]](#)

[Logout \[page 863\]](#)

[Set the Default Command Context \[page 864\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

[Set the Default Command Context \[page 864\]](#)

### 5.1.2.2.4 Login

Login with the CLI for SAP Cloud Platform (sapcp CLI) is on global account level.

#### Prerequisites

- Your global account is on SAP Cloud Platform Feature Set B. See [Cloud Management Tools — Feature Set Overview](#).
- To log in to your global account, you need the following:
  - The URL of the CLI server. If you're using the trial version, the CLI will propose the correct URL upon login.
  - The subdomain of your global account. You find the global account subdomain in the cockpit in the global account view or under [Switch Global Account](#).

#### Context

Logging in to SAP Cloud Platform with the command-line interface creates a token that is stored on your computer and allows to close and re-open the command line without losing your login. With each command call, this token is renewed and valid for 24 hours. So if you take a longer break from working with the sapcp CLI, you will have to log in again. If you want to end your login earlier, you can use `sapcp logout`.

#### Procedure

Use `sapcp login` and, optionally, provide the required information as parameters. Note that the sapcp CLI prompts for all parameters if you don't provide them.

Usage: `sapcp [OPTIONS] login [PARAMS]`

## Parameters

--url <URL>

The CLI server URL.

### i Note

The client proposes the default server for the trial version of SAP Cloud Platform (<https://cpcli.cf.eu10.hana.ondemand.com>), which you can confirm by pressing **ENTER**. If you want to use a different server, you have to specify it here.

---

--subdomain <GLOBALACCOUNT>

The subdomain of the global account you want to log in to. You should have obtained the subdomain from your operator; but you can also find it in the cockpit in the global account view or under *Switch Global Account* dialog.

### i Note

If you don't find the subdomain of the global account in your cockpit, ensure that you are using a global account on SAP Cloud Platform Feature Set B. See [Cloud Management Tools — Feature Set Overview](#).

---

--user <USER>

Your user name, usually an email address.

---

--password <PASSWORD>

Your password.

### → Tip

We do not 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.

## ↳ Sample Code

```
sapcp login --url https://cpcli.cf.eu10.hana.ondemand.com --subdomain my-global-account --user name@example.com --password my-password
```

The CLI server URL for the trial version of SAP Cloud Platform and, if you've logged in before, the subdomain, and user from the last login will be suggested. You can then press **Enter** to confirm, or type in different values.

## ↳ Output Code

```
CLI server URL [https://cpcli.cf.eu10.hana.ondemand.com]>
Subdomain [my-global-account]>
User [name@example.com]>
```

## Results

Upon successful login, the sapcp CLI creates a folder (sapcp) and a configuration file (config.json) in the default location of your user data directory:

- Microsoft Windows: C:\Users\<username>\AppData\Local\SAP\sapcp\config.json
- Apple macOS / Linux: \$HOME/.sapcp/config.json

To change this location, use the SAPCP\_CLIENTCONFIG environment variable or the --config option, which you can append to each command call. See [Specify the Location of the Configuration File \[page 866\]](#).

→ Tip

You've logged in to the global account and all commands are executed in the context of this global account. To change this default context for subsequent commands to a subaccount or directory of this global account, use sapcp target. See [Set the Default Command Context \[page 864\]](#).

**Task overview:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[View Version and Current Context \[page 860\]](#)

[Logout \[page 863\]](#)

[Set the Default Command Context \[page 864\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

### 5.1.2.2.5 Logout

Logging out of the configured server removes all user-specific data from the configuration file.

## Context

Once you're finished using the sapcp 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 sapcp CLI will propose your current subdomain and user so you won't have to type it in again.

## Procedure

To log out, use `sapcp 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.

**Task overview:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[View Version and Current Context \[page 860\]](#)

[Login \[page 861\]](#)

[Set the Default Command Context \[page 864\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

[Login \[page 861\]](#)

### 5.1.2.2.6 Set the Default Command Context

Change the default context for all command calls to the global account, a directory, or a subaccount by using the `sapcp target` command.

## Context

Targeting works along the hierarchy of your account model:

- After login, the global account is targeted by default.
- When targeting a subaccount or directory, you can execute commands in its parent directory or global account by using '-dir' or '-ga' without a value.
- Commands that only work on a directory or global account level will be executed in the parent directory or global account of the current target.

## Procedure

Use `sapcp target [PARAMS]` to set the context for subsequent commands.

Usage: `sapcp [OPTIONS] target [PARAMS]`

## Parameters

--global-account, -ga <SUBDOMAIN>	Only the global account of the active login can be targeted. As only one active login is possible, SUBDOMAIN can be omitted.
--directory, -dir<ID>	The ID of the directory to be targeted.
--subaccount, -sa <ID>	The ID of the subaccount to be targeted. You can find this ID by using sapcp list accounts/subaccount.

## Results

Once you have set the context to a subaccount, all subsequent commands are executed there, unless you specify a different one by providing one of the parameters directly in a command call.

### → Tip

To find out your current context, use sapcp --info or simply sapcp.

## Example

To set the context back to the global account, use sapcp target -ga.

**Task overview:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[View Version and Current Context \[page 860\]](#)

[Login \[page 861\]](#)

[Logout \[page 863\]](#)

[Specify the Location of the Configuration File \[page 866\]](#)

[Commands in the sapcp CLI \[page 867\]](#)

[Global Accounts](#)

## 5.1.2.2.7 Specify the Location of the Configuration File

You can change the location of the configuration file by using the `--config` option.

### Context

Upon successful login, the sapcp CLI creates a folder (`sapcp`) and a configuration file (`config.json`) in the default location of your user data directory:

- Microsoft Windows: `C:\Users\<username>\AppData\Local\SAP\sapcp\config.json`
- Apple macOS / Linux: `$HOME/.sapcp/config.json`

This folder serves as the working directory of the sapcp 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:

```
sapcp --config "<file path>" login
```

2. Specify this location either with the `SAPCP_CLIENTCONFIG` environment variable, or use the `--config` option with each subsequent command call.

#### ↳ Sample Code

```
sapcp --config "<file path>"
```

### 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	Terminal B
1. Log in to SAP Cloud Platform using the default location of the configuration file:	1. Log in to SAP Cloud Platform using a different location of the configuration file:
<pre>sapcp login</pre>	<pre>sapcp --config "C:\my-cli-folder" login</pre>
Run all commands as usual. The sapcp CLI uses the default configuration file.	Use this option with each command call.
2. Set the default context to subaccount 1000:	2. Set the default context to subaccount 2000:
<pre>sapcp target --subaccount 1000</pre>	<pre>sapcp --config "C:\my-cli-folder" target --subaccount 2000</pre>
3. List the users of subaccount 1000:	3. List the users of subaccount 2000:
<pre>sapcp list security/user</pre>	<pre>sapcp --config "C:\my-cli-folder" list security/user</pre>

**Task overview:** [General Commands and Options in the sapcp CLI \[page 854\]](#)

## Related Information

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[View Version and Current Context \[page 860\]](#)

[Login \[page 861\]](#)

[Logout \[page 863\]](#)

[Set the Default Command Context \[page 864\]](#)

### 5.1.2.3 Commands in the sapcp CLI

A list of all tasks and respective commands that are available in the CLI for SAP Cloud Platform (sapcp CLI).

#### → Tip

After login, all commands are executed in the global account. If you want to work in a specific subaccount or a directory, we recommend to set the default context by using `sapcp target` (see [Set the Default Command Context \[page 864\]](#)). Once you have done so, you do not have to pass the subaccount or directory parameter with every command.

## → Tip

You can find extensive help about each command directly in the sapcp CLI by using the `--help` option. For example, use one of the following commands for help on `sapcp list accounts/subaccount`:

```
sapcp --help list accounts/subaccount
```

```
sapcp list accounts/subaccount --help
```

### [Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

Use the CLI for SAP Cloud Platform (sapcp CLI) to manage operations with global accounts, directories, and subaccounts.

### [Setting Entitlements Using the sapcp CLI \[page 870\]](#)

Use the CLI for SAP Cloud Platform (sapcp CLI) to set entitlements to define the functionality or permissions available for users of global accounts, directories, and subaccounts.

### [Working with Environments Using the sapcp CLI \[page 871\]](#)

Use the CLI for SAP Cloud Platform (sapcp CLI) to get details, or to create or delete 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 sapcp CLI \[page 872\]](#)

Use the CLI for SAP Cloud Platform (sapcp CLI) to manage the multitenant applications to which a subaccount is entitled to subscribe.

### [Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

Use the CLI for SAP Cloud Platform (sapcp CLI) to get details, or to create or delete resource provider instances in a global account.

### [Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

### [Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

Use the CLI for SAP Cloud Platform to perform various operations related to your platforms, attached service brokers, service instances, and service bindings.

## Related Information

[Login \[page 861\]](#)

[Command Syntax \[page 855\]](#)

[Get Help \[page 859\]](#)

[Set the Default Command Context \[page 864\]](#)

### 5.1.2.3.1 Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp CLI) to manage operations with global accounts, directories, and subaccounts.

#### Working with Global Accounts

Task	Run the command...
Get details about a global account, and the account structure (directories and subaccounts) of the global account.	sapcp get accounts/global-account
Update a global account	sapcp update accounts/global-account

For more information, see [Global Accounts](#).

#### Working with Directories (Beta)

Directories allow you to organize and manage your subaccounts according to your technical and business needs.

##### → Remember

Beta features may be changed by SAP at any time for any reason without notice. Do not use beta functionality with subaccounts that are used in a production environment. SAP shall not be liable for any errors or damage arising from the use of such features.

Task	Run the command...
Get details about a directory and list the subaccounts in the directory	sapcp get accounts/directory
Create a directory	sapcp create accounts/directory
Update a directory	sapcp update accounts/directory
Delete a directory	sapcp delete accounts/directory
Enable additional features in a directory	sapcp enable accounts/directory
List the custom properties assigned to a directory	sapcp list accounts/custom-property

For more information, see [Directories \(Beta\) \[Feature Set B\]](#).

## Working with Subaccounts

Task	Run the command...
List all subaccounts in a global account	sapcp list accounts/subaccount
Get details about a subaccount	sapcp get accounts/subaccount
Create a subaccount	sapcp create accounts/subaccount
Update a subaccount	sapcp update accounts/subaccount
Delete a subaccount	sapcp delete accounts/subaccount
Move a subaccount (beta)	sapcp move accounts/subaccount
List the custom properties assigned to a subaccount	sapcp list accounts/custom-property
Get all available regions for global account	sapcp list accounts/available-region

For more information, see [Subaccounts](#).

**Parent topic:** Commands in the sapcp CLI [page 867]

## Related Information

- [Setting Entitlements Using the sapcp CLI \[page 870\]](#)
- [Working with Environments Using the sapcp CLI \[page 871\]](#)
- [Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)
- [Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)
- [Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)
- [Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)
- [Set the Default Command Context \[page 864\]](#)
- [Custom Properties \[Feature Set B\]](#)

### 5.1.2.3.2 Setting Entitlements Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp CLI) to set entitlements to define the functionality or permissions available for users of global accounts, directories, and subaccounts.

Task	Run the command...
Get all the entitlements and quota assignments for a global account, directories, and subaccounts	sapcp list accounts/entitlement
Assign or update an entitlement to a subaccount or a directory	sapcp assign accounts/entitlement

**Parent topic:** Commands in the sapcp CLI [page 867]

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Working with Environments Using the sapcp CLI \[page 871\]](#)

[Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)

[Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

[Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

[Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

### 5.1.2.3.3 Working with Environments Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp CLI) to get details, or to create or delete environment instances in a subaccount. For example, enable the Cloud Foundry environment by creating a Cloud Foundry org (environment instance).

Task	Run the command ...
Get all available environments for a subaccount	<code>sapcp list accounts/available-environment</code>
Get all environment instances of a subaccount	<code>sapcp list accounts/environment-instance</code>
Get a specific environment instance of a subaccount	<code>sapcp get accounts/environment-instance</code>
Create an environment instance in a subaccount	<code>sapcp create accounts/environment-instance</code>
Delete environment instances of a subaccount	<code>sapcp delete accounts/ environment-instance</code>

**Parent topic:** Commands in the sapcp CLI [page 867]

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Setting Entitlements Using the sapcp CLI \[page 870\]](#)

[Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)

[Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

[Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

[Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

## 5.1.2.3.4 Working with Multitenant Applications Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp CLI) to manage the multitenant applications to which a subaccount is entitled to subscribe.

Task	Run the command...
Get details of a multitenant application in a subaccount	<code>sapcp get accounts/subscription</code>
Get all applications to which a subaccount is entitled to subscribe	<code>sapcp list accounts/subscription</code>
Subscribe to an application from a subaccount	<code>sapcp subscribe accounts/subaccount</code>
Unsubscribe an application from a subaccount	<code>sapcp unsubscribe accounts/subaccount</code>

**Parent topic:** [Commands in the sapcp CLI \[page 867\]](#)

### Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Setting Entitlements Using the sapcp CLI \[page 870\]](#)

[Working with Environments Using the sapcp CLI \[page 871\]](#)

[Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

[Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

[Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

## 5.1.2.3.5 Working with External Resource Providers Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp CLI) to get details, or to create or delete resource provider instances in a global account.

### i 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 Cloud Platform.

Task	Run the command...
List all resource provider instances in a global account	<code>sapcp list accounts/resource-provider</code>
Get details about a resource provider instance	<code>sapcp get accounts/resource-provider</code>
Create a resource provider instance	<code>sapcp create accounts/resource-provider</code>
Update a resource provider instance	<code>sapcp update accounts/resource-provider</code>
Delete a resource provider instance	<code>sapcp delete accounts/resource-provider</code>

For more information, see [Managing Resource Providers \[page 833\]](#).

**Parent topic:** [Commands in the sapcp CLI \[page 867\]](#)

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Setting Entitlements Using the sapcp CLI \[page 870\]](#)

[Working with Environments Using the sapcp CLI \[page 871\]](#)

[Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)

[Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

[Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

### 5.1.2.3.6 Managing Users and Their Authorizations Using the sapcp CLI

Use the CLI for SAP Cloud Platform (sapcp 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. You can set one of these as the default context using the `sapcp target` command. See [Set the Default Command Context \[page 864\]](#).

## 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 \[page 826\]](#).

Task	Run the command ...
List all apps of the global account	sapcp list security/app
Get details about a specific application	sapcp get security/app
List all role templates of an application	sapcp list security/role-template
Get details about a role template	sapcp get security/role-template
List all roles	sapcp list security/role
List all roles of a specific application	sapcp list security/role
List all roles of a specific role template	sapcp list security/role
Get details about a specific role	sapcp get security/role
Create a role	sapcp create security/role
Delete a role	sapcp delete security/role
Add a role to a role collection	sapcp add security/role
Remove a role from a role collection	sapcp remove security/role

## Managing Role Collections

Role collections are user-related authorizations that restrict access to resources and services based on defined user permissions. They consist of individual roles, which, in turn, are based on role templates. There are four predefined role collections: Global Account Administrator, Subaccount Administrator, Global Account Viewer, and Subaccount Viewer. For more information, see [Building Roles and Role Collections for Applications \[page 819\]](#).

Task	Run the command ...
List all role collections	sapcp list security/role-collection
Get details about a specific role collection	sapcp get security/role-collection
Create a role collection	sapcp create security/role-collection
Change the description of a role collection	sapcp update security/role-collection
Delete a role collection	sapcp delete security/role-collection

## Managing Users and Assigning Role Collections

You can display all users of the global account and their assignment to role collections. You can also assign and unassign role collections to and from users. See [Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#).

Task	Run the command ...
List all users	<code>sapcp list security/user</code>
Get details about a specific user, including role collections	<code>sapcp get security/user</code>
Delete a user	<code>sapcp delete security/user</code>
Assign a role collection to a user	<code>sapcp assign security/role-collection</code>
Unassign a role collection from a user	<code>sapcp unassign security/role-collection</code>

**Parent topic:** Commands in the sapcp CLI [page 867]

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

[Setting Entitlements Using the sapcp CLI \[page 870\]](#)

[Working with Environments Using the sapcp CLI \[page 871\]](#)

[Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)

[Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)

[Working With the Service Management Resources Using the sapcp CLI \[page 875\]](#)

[Security Administration: Managing Authentication and Authorization \[page 784\]](#)

[Set the Default Command Context \[page 864\]](#)

[Role Collections and Roles in Global Accounts and Subaccounts \[Feature Set B\]](#)

### 5.1.2.3.7 Working With the Service Management Resources Using the sapcp CLI

Use the CLI for SAP Cloud Platform 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 runtime platform.

For more information about the Service Management, see [Service Management Overview](#).

#### → Tip

All of these commands are executed for a specific subaccount. We recommend using the `sapcp target` command to set the default context to a subaccount. See [Set the Default Command Context \[page 864\]](#).

## Managing Platforms

Task	Run the command ...
List all registered platforms in the current subaccount	<code>sapcp list services/platform</code>
Get details about a specific platform registered in the current subaccount	<code>sapcp get services/platform</code>
Register a new platform in the current subaccount	<code>sapcp register services/platform</code>
Unregister an existing platform in the current subaccount	<code>sapcp unregister services/platform</code>

## Managing Service Brokers

Task	Run the command ...
List all registered brokers in the current subaccount	<code>sapcp list services/broker</code>
Register a new service broker in the current subaccount	<code>sapcp register services/broker</code>
Update an existing service broker in the current subaccount	<code>sapcp update services/broker</code>
Unregister an existing service broker in the current subaccount	<code>sapcp unregister services/broker</code>

## Managing Service Instances

Task	Run the command ...
List all service instances associated with the current subaccount.	<code>sapcp list services/instance</code>
Get details about a specific service instance associated with the current subaccount.	<code>sapcp get services/instance</code>
Create a new service instance of the service you wish to consume.	<code>sapcp create services/instance</code>
Delete an existing service instance.	<code>sapcp delete services/instance</code>

## Managing Service Bindings

Task	Run the command ...
List all service bindings associated with the current subaccount.	<code>sapcp list services/binding</code>
Get details about a specific service binding associated with the current subaccount.	<code>sapcp get services/binding</code>
Create a new binding between an existing service instance and an application.	<code>sapcp create services/binding</code>
Delete an existing binding between a service instance and an application.	<code>sapcp delete services/instance</code>

## Viewing Service Plans

Task	Run the command ...
List all service plans available for consumption of services associated with your current subaccount.	<code>sapcp list services/plan</code>
Get details about a specific service plan available for consumption of a service associated with your current subaccount.	<code>sapcp get services/plan</code>

## Viewing Service Offerings

Task	Run the command ...
List all service offerings associated with your current subaccount.	<code>sapcp list services/offering</code>
Get details about a specific service offering associated with your subaccount.	<code>sapcp get services/offering</code>

[Parent topic: Commands in the sapcp CLI \[page 867\]](#)

## Related Information

[Working with Global Accounts, Directories, and Subaccounts Using the sapcp CLI \[page 869\]](#)

- [Setting Entitlements Using the sapcp CLI \[page 870\]](#)
- [Working with Environments Using the sapcp CLI \[page 871\]](#)
- [Working with Multitenant Applications Using the sapcp CLI \[page 872\]](#)
- [Working with External Resource Providers Using the sapcp CLI \[page 872\]](#)
- [Managing Users and Their Authorizations Using the sapcp CLI \[page 873\]](#)

## 5.1.2.4 Troubleshooting and Support

Troubleshooting and support information for the CLI for SAP Cloud Platform (sapcp CLI).

If you have trouble logging in or get started:

### Make sure your global account is on feature set B

The CLI is part of our cloud management tools feature set B. This means that you can only access a global account that uses this features set. For example, Cloud Foundry trial accounts that were created after April 2, 2020 use feature set B. For more information, see [Cloud Management Tools — Feature Set Overview](#).

### Follow the workflow 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 via the Command Line \[Feature Set B\] \[page 17\]](#) or [Setting Up a Global Account via the Command Line \[Feature Set B\] \[page 25\]](#).

If you run into problems using the sapcp CLI, try the following:

### Make sure that you are using a recent version of the CLI client

We recommend checking for updates of the CLI 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 `sapcp` or `sapcp --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 `sapcp --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.

## Start --verbose mode

If you get an error message that is not helpful, or if you want to find out more about what exactly happened after you ran a command, use the `sapcp --verbose` option with your command and run it again. The `sapcp` 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 `sapcp list security/role-collection` command call without the option:

### ↳ Sample Code

```
sapcp 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 `sapcp --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.

### ↳ Sample Code

```
sapcp --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/.sapcp/config.json
2019/12/18 10:35:21.309386 login.go:138: Reading configuration from &{/Users/my-user/Library/Caches/.sapcp 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/.sapcp/config.json
2019/12/18 10:35:21.309603 login.go:138: Reading configuration from &{/Users/my-user/Library/Caches/.sapcp 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
```

## Get 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 \[page 1161\]](#).

### 5.1.3 Account Administration Using APIs (Beta) [Feature Set B]

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

Provides information about using the APIs of the SAP Cloud Platform Cloud Management service to manage some of the administrative operations in your SAP Cloud Platform accounts.

#### i Note

This is a beta feature.

Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. This means that beta features may be changed by SAP at any time for any reason without notice. Beta features aren't for productive use. Any use of beta functionality is at your own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

The core service APIs allow you to manage, build, and extend the core capabilities of SAP Cloud Platform. Each core service focuses on a different aspect of the platform, from the management of accounts and product entitlements, through to the registration and provisioning of subscriptions for multitenant applications and services.

Go to <https://accounts-service.<app domain>.<landscape domain>/swagger-index.html>.

### • Example

If your account is running on the Europe (Frankfurt) region, use this URL:

<https://accounts-service.cfapps.eu10.hana.ondemand.com/swagger-index.html>

### → Remember

To call the core platform API methods, you must obtain an access token. See [Manage Authentication and Authorization Process for Cloud Management APIs \[page 881\]](#).

## Related Information

[Error Response Format \[page 908\]](#)

[Asynchronous Jobs \[page 910\]](#)

[Rate Limiting \[page 910\]](#)

[Using Platform APIs](#)

[Cloud Management Tools — Feature Set Overview](#)

### 5.1.3.1 Manage Authentication and Authorization Process for Cloud Management APIs

Core service APIs of SAP Cloud Platform are protected with the OAuth 2.0 Password Grant type. This procedure guides you through the steps to create an OAuth client and obtain an access token from the `xsuaa` service instance to call the SAP Cloud Platform Cloud Management service APIs.

#### i Note

For information about getting an application access token for the SaaS Provisioning service APIs, see [Get an Application Access Token \[page 209\]](#) and [Cloud Management - Service Plans \[page 885\]](#).

## Prerequisites

Your global account admin has entitled at least one service plan from the Cloud Management service in your subaccount. See [Cloud Management - Service Plans \[page 885\]](#) and [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).

#### i Note

The service plan for the SaaS Manager APIs is entitled by default to all subaccounts.

## Procedure

1. Create a service instance for the Cloud Management service (cis).

When you create the service instance, specify the following parameters:

- PLAN: The name of the service plan you want to use. See [Cloud Management - Service Plans \[page 885\]](#).
- 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 the Cloud Foundry environment using the SAP Cloud Platform cockpit or the cf CLI, see [Creating Service Instances \[page 372\]](#).
- To create a service instance in other environments using the Service Management APIs, see [Accessing the API](#), and then [Create a Service Instance](#).
- To create a service instance in Kyma using the Kyma dashboard, see [Create Service Instances Using the Kyma Console UI \[page 638\]](#).

### → Recommendation

If you are not working in Cloud Foundry, Kyma, or Kubernetes, use the Service Management 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 following parameters:

- SERVICE\_INSTANCE: Name of the service instance for which to create the service key.
- SERVICE\_KEY: Name for the service key.

There are several options available to create instances depending on the environment you use.

- To create a service key in the Cloud Foundry environment using the SAP Cloud Platform cockpit or the cf CLI, see [Creating Service Keys \[page 379\]](#).
- To create credentials for calling the service and retrieving information in the Kyma environment, see [Creating Credentials \[page 644\]](#).

### ⇐ 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:

```
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:

```
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>'
```

#### i Note

If two-factor authentication (2FA) is activated on the SAP Cloud Platform 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`.

#### i 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": "<xappname>.job.read <xappname>.event.read"
}
```

See [Enable API Access to an XSUAA Configuration](#).

4. Call the Cloud Management APIs using Swagger 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 example for a service instance in Cloud Foundry.
  2. Browse to `<endpoint url>/swagger-ui.html`.
  3. In Swagger, choose [Authorize](#).
  4. Enter the `api_key` and choose [Authorize](#).

**i 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!*.

## Related Information

[Get API Access \[page 912\]](#)

## 5.1.3.1.1 Cloud Management - Service Plans

Describes the service plans for the SAP Cloud Platform Cloud Management (cis) service and the scopes they provide.

Service Display Name	Technical Name	Service Plan	Scopes
Cloud Management	cis	<b>central:</b> Service plan for using Cloud Management APIs to manage your global accounts, subaccounts, directories, and entitlements.	<ul style="list-style-type: none"><li>• global-account.read</li><li>• global-account.update</li><li>• global-account.subaccount.read</li><li>• global-account.subaccount.create</li><li>• global-account.subaccount.update</li><li>• global-account.subaccount.delete</li><li>• global-account.account-directory.read</li><li>• global-account.account-directory.create</li><li>• global-account.account-directory.update</li><li>• global-account.account-directory.delete</li><li>• global-account.entitlement</li><li>• global-account.entitlement.subaccount.update</li><li>• global-account.region.read</li><li>• catalog.product.update</li><li>• catalog.product.delete</li><li>• directory.entitlement.update</li><li>• directory.entitlement.read</li><li>• job.read</li><li>• cis-central.event.read</li></ul>

Service Display Name	Technical Name	Service Plan	Scopes
Cloud Management		<b>local:</b> Service plan for using Cloud Management APIs to manage your environments and subscriptions to multi-tenant applications.	<ul style="list-style-type: none"> <li>subaccount.entitlement.read</li> <li>subaccount.environment.read</li> <li>subaccount.environment.create</li> <li>subaccount.environment.delete</li> <li>subaccount.application.subscription.read</li> <li>subaccount.application.subscription.update</li> <li>job.read</li> <li>cis-local.event.read</li> </ul>
SaaS Provisioning	saas-registry	<b>application:</b> Service plan for application owners to manage the lifecycle of multi-tenant applications with SaaS Manager APIs.	<ul style="list-style-type: none"> <li>job.read</li> <li>subscription.read</li> <li>subscription.write</li> <li>entitlement.read</li> </ul>

### 5.1.3.2 Manage Authentication and Authorization Process for Resource Consumption APIs

Platform APIs of SAP Cloud Platform are protected with OAuth 2.0 client credentials. This procedure guides you through the steps to create an OAuth client and obtain an access token to call the SAP Cloud Platform Usage Data Management APIs.

#### Prerequisites

- You have downloaded and installed the latest version of the Cloud Foundry command line interface (cf CLI). See [Download and Install the Cloud Foundry Command Line Interface](#).
- To use the Usage Data Management APIs, you must have service access to the relevant **uas** service plan. For information about available service plans, see [Usage Data Management - Service Plans \[page 889\]](#).

#### → Tip

To check if you have service access to these service plans, you can use the Cloud Foundry marketplace. Run these commands in the cf CLI:

```
cf marketplace -s uas
```

You should see the service plans to which your orgs have access.

If you do not have access to the relevant service plans, make sure that the global account admin has entitled them in your subaccount. See [Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#).

- You have created a space in your Cloud Foundry org.

## Procedure

1. Log in to your space using of CLI. See [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface](#).

**Example:**

```
cf api https://api.<landscape domain>
cf login
cf target -o ORG -s SPACE
```

2. Using the cf CLI, create a service instance for the service plans that correspond to the APIs that you want to use.

```
cf create-service uas PLAN SERVICE_INSTANCE
```

Specify the following parameters:

- **PLAN**: The name of the service plan that you want to use. See [Usage Data Management - Service Plans \[page 889\]](#).
- **SERVICE\_INSTANCE**: Name of the service instance.

See [Create Service Instances Using the Cloud Foundry Command Line Interface](#).

3. Get the OAuth 2.0 client information and `uas` service endpoints using either one of the following methods:

1. Bind the service instance to your application. See [Bind Service Instances to Applications Using the Cloud Foundry Command Line Interface](#).

- Display the environment of your application.

```
cf env APP-NAME
```

- Locate the details of your `uas` service instance in the application environment, for example:

```
System-Provided:
{
  "VCAP_SERVICES": {
    ...
    "uas": [
      {
        "credentials": {
          "clientid": <client_id>,
          "clientsecret": <client_secret>,
          "url": <uaa_url>,
          "target_url": <endpoint url>
        }
      },
      "instance_name": SERVICE_INSTANCE,
      "plan": "reporting-ga-admin"
    ]
  ...
}
```

2. Create a service key. See [Create Service Keys Using the Cloud Foundry Command Line Interface](#).

```
cf create-service-key SERVICE_INSTANCE SERVICE_KEY
```

Specify the following parameters:

- SERVICE\_INSTANCE: Name of the service instance.
- SERVICE\_KEY: Name of the service key.

Display the service key information:

```
cf service-key SERVICE_INSTANCE SERVICE_KEY
{
  "credentials": {
    "clientid": <client_id>,
    "clientsecret": <client_secret>,
    "url": <uaa_url>,
    "target_url": <endpoint_url>
  },
  "instance_name": SERVICE_INSTANCE,
  "plan": "reporting-ga-admin"
}
```

4. Use uaa\_url, clientid, and clientsecret to request an access token using the following command:

```
curl '<uaa_url>/oauth/token' -X POST \
-H 'Accept: application/json' \
-d
'grant_type=client_credentials&client_id=<clientid>&client_secret=<clientsecret>'
```

### i 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": "<xappname>.UAS.reporting.GA_Admin"
}
```

See [Enable API Access to an XSUAA Configuration](#).

5. Call the Cloud Management APIs using Swagger with the<access\_token> that you received in the step 4:
  1. Choose one of the endpoints URLs of your service instance from step 3a or step 3b.
  2. Browse to <endpoint url>/swagger-ui.html.
  3. Choose **Authorize**.
  4. Enter the api\_key and choose **Authorize**.

### i Note

Use the <access\_token> that you received in step 4 to compose the api\_key in the format bearer <access\_token>.

6. Choose the API that you want to call, provide the relevant parameters, and choose **Try it out!**

### 5.1.3.2.1 Usage Data Management - Service Plans

Describes the SAP Cloud Platform Usage Data Management service plans.

#### i Note

The Usage Data Management APIs can be found in the [Resource Consumption](#) tile in Swagger.

Display Name	Technical Service Name	Service Plans	Scopes
Usage Data Management (UDM)	uas	reporting-ga-admin: Service plan for using the Usage Data Management service APIs. This microservice is responsible for generating reports based on the resource and cost consumption.	Reporting.GA_Admin

### 5.1.3.3 Monitoring Usage Information Using APIs (Beta)

#### i Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

Provides information about using the APIs of the SAP Cloud Platform Usage Data Management service 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.

#### i Note

This is a beta feature.

Beta features aren't part of the officially delivered scope that SAP guarantees for future releases. This means that beta features may be changed by SAP at any time for any reason without notice. Beta features aren't for productive use. Any use of beta functionality is at your own risk, and SAP shall not be liable for errors or damages caused by the use of beta features.

The service accumulates the information and provides reports in several business systems (reporting and operations) for resource planning and cross-billing purposes.

#### i Note

The Usage Data Management service is not sold to SAP Cloud Platform customers as a service; however, customers can see their usage data in the SAP Cloud Platform cockpit, for example, in the Usage Analytics pages.

## Accessing the API

Go to `https://uas-reporting.<app domain>.<landscape domain>/swagger-ui.html`.

### ❖ Example

If your account is running on the Europe (Frankfurt) region, use this URL:

```
https://uas-reporting.cfapps.eu10.hana.ondemand.com/swagger-ui.html
```

### → Remember

To call the core platform API methods, you must obtain an access token. See [Manage Authentication and Authorization Process for Resource Consumption APIs \[page 886\]](#).

**Base URI:** `https://uas-reporting.cfapps.<landscape domain>/reports/v1`

Your global account admin has entitled the service plan for the Usage Data Management service in your subaccount. See [Usage Data Management - Service Plans \[page 889\]](#).

## Methods

HTTP Method	Action	URI
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

### 5.1.3.4 Using the Events Service APIs

The Events service provides REST APIs that collect information about events relating to account administrative operations in the SAP Cloud Platform Cloud Management services, such as Accounts, Entitlements, Provisioning, and SaaS Manager, 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 SaaS Manager service to send emails notifying when applications are subscribed to from a subaccount.

**Permissions:** A service instance of the Cloud Management service is required. The scopes required to get the events are available for all of the Cloud Management service plans. See [Cloud Management - Service Plans \[page 885\]](#).

For more information about permissions, see [Manage Authentication and Authorization Process for Cloud Management APIs \[page 881\]](#).

**Base URI:** `https://events-service.<app domain>.<landscape domain>`

## Methods

HTTP Method	Action	URI
GET	Get events	<code>events-service.&lt;app domain&gt;.&lt;landscape domain&gt;/cloud-management/v1/events</code>
GET	Get event types	<code>events-service.&lt;app domain&gt;.&lt;landscape domain&gt;/cloud-management/v1/events/types</code>

## Related Information

[Manage Authentication and Authorization Process for Cloud Management APIs \[page 881\]](#)

### 5.1.3.4.1 Get Events

Get all events associated with administrative operations in your SAP Cloud Platform accounts.

The events you get depend on the scopes you used to access the API.

To learn more about the scopes, see [Cloud Management - Service Plans \[page 885\]](#).

## 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 903]</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_Creation, SubaccountAppSubscription</li> </ul>

Parameter Name	Required	Parameter Type	Description
			scription_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>
id	No	Array of integers	The ID of the event.
pageNum	No	Integer	The page number to retrieve.
pageSize	No	Integer	The number of events to retrieve per page (max = 150).
searchParams	No	JSON Object	Additional search parameters that depend on the type of the events.
sortField	No	String	Field by which to sort the events.
sortOrder	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>
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>

## 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	<p>Array of JSON objects:</p> <p><code>BusinessEventResponseObject</code></p>	<p>List of the event objects.</p> <p>See the next table <code>BusinessEventResponseObject</code> for its parameters.</p>
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, AccountDirectoryTenant_Creation, AccountDirectoryTenant_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\"},\"mongodbmedium\":{\"decreased\":{},\"increased\":{\"6699b187-d17c-4783-bc8c-3263690d4aac\":{\"amount\":1,\"servicePlanAssignmentId\":21590}},\"serviceName\":\"mongodb\",\"servicePlanName\":\"medium\"},"
            }
        }
    ]
}

```

```

\"servicePlanUniqueIdentifier\":\"mongodbmedium\", \"isEntitlementUnlimited\":
\":false, \"assignmentLevel\":\"SUBACCOUNT\"}, \"applicationruntime\":{\"decreased\":
{}, \"increased\":[\"6699b187-d17c-4783-bc8c-3263690d4aac\"]: {\"amount\":
100, \"servicePlanAssignmentId\":21591}}, \"serviceName\":\"APPLICATION_RUNTIME\",
\"servicePlanName\":\"MEMORY\", \"servicePlanUniqueIdentifier\":
\"applicationruntime\", \"isEntitlementUnlimited\":false, \"assignmentLevel\":
\"SUBACCOUNT\"}, \"cis-local\":[{\"decreased\":[{}], \"increased\":[\"6699b187-
d17c-4783-bc8c-3263690d4aac\"]: {\"amount\":5, \"servicePlanAssignmentId\":21592}},
\"serviceName\":\"cis\", \"servicePlanName\":\"local\",
\"servicePlanUniqueIdentifier\":\"cis-local\", \"isEntitlementUnlimited\":false,
\"assignmentLevel\":\"SUBACCOUNT\"}, \\"html5-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\"}, \\"rabbitmqsmall\":[{\"decreased\":[{}], \"increased\":[\"6699b187-d17c-4783-
bc8c-3263690d4aac\"]: {\"amount\":1, \"servicePlanAssignmentId\":21594}},
\"serviceName\":\"rabbitmq\", \"servicePlanName\":\"small\",
\"servicePlanUniqueIdentifier\":\"rabbitmqsmall\", \"isEntitlementUnlimited\":
false, \"assignmentLevel\":\"SUBACCOUNT\"}, \\"rabbitmq-virtualhost\":
{\"decreased\":[{}], \"increased\":[\"6699b187-d17c-4783-bc8c-3263690d4aac\"]:
{\"amount\":5, \"servicePlanAssignmentId\":21595}}, \"serviceName\":\"rabbitmq\",
\"servicePlanName\":\"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\"}, \\"postresqllarge\": {\"decreased\":
{}, \"increased\":[\"6699b187-d17c-4783-bc8c-3263690d4aac\"]: {\"amount\":
1, \"servicePlanAssignmentId\":21597}}, \"serviceName\":\"postresql\",
\"servicePlanName\":\"large\", \"servicePlanUniqueIdentifier\":\"postresqllarge\",
\"isEntitlementUnlimited\":false, \"assignmentLevel\":
\"SUBACCOUNT\"}, \\"screeninghits-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\",
\"servicePlanName\":\"dev\", \"servicePlanUniqueIdentifier\":\"redis-dev\",
\"isEntitlementUnlimited\":false, \"assignmentLevel\":
\"SUBACCOUNT\"}, \\"redislarge\":[{\"decreased\":[{}], \"increased\":[\"6699b187-d17c-4783-bc8c-3263690d4aac\"]:
{\"amount\":1, \"servicePlanAssignmentId\":21600}], \"serviceName\":\"redis\",
\"servicePlanName\":\"large\", \"servicePlanUniqueIdentifier\":\"redislarge\",
\"isEntitlementUnlimited\":false, \"assignmentLevel\":
\"SUBACCOUNT\"}, \\"rabbitmq-dev\":[{\"decreased\":[{}], \"increased\":[\"6699b187-d17c-4783-
bc8c-3263690d4aac\"]: {\"amount\":10, \"servicePlanAssignmentId\":21601}],
\"serviceName\":\"rabbitmq\", \"servicePlanName\":\"dev\",
\"servicePlanUniqueIdentifier\":\"rabbitmq-dev\",
\"isEntitlementUnlimited\":false, \"assignmentLevel\":
\"SUBACCOUNT\"}, \\"mongodblarge\":[{\"decreased\":[{}], \"increased\":[\"6699b187-d17c-4783-
bc8c-3263690d4aac\"]: {\"amount\":1, \"servicePlanAssignmentId\":21602}},
\"serviceName\":\"mongodb\", \"servicePlanName\":\"large\",
\"servicePlanUniqueIdentifier\":\"mongodblarge\", \"isEntitlementUnlimited\":
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\":\"feature-flags\", \"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\":
\"auditlog-viewer_standard_\"

```

```

\":\"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\"},\"postgresqlxsmall\":{\"decreased\":{},\"increased\":{\"6699b187-d17c-4783-bc8c-3263690d4aac\":{\"amount\":1,\"servicePlanAssignmentId\":21616}},\"serviceName\":\"postgresql\",\"servicePlanName\":\"xsmall\",\"servicePlanUniqueIdentifier\":\"postgresqlxsmall\",\"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\"},\"postgresql-dev\":{\"decreased\":{},\"increased\":{\"6699b187-d17c-4783-bc8c-3263690d4aac\":{\"amount\":150,\"servicePlanAssignmentId\":21619}},\"serviceName\":\"postgresql\",\"servicePlanName\":\"dev\",\"servicePlanUniqueIdentifier\":\"postgresql-dev\",\"isEntitlementUnlimited\":false,\"assignmentLevel\":\"SUBACCOUNT\"},\"sample-saas-app-i337243-test\":{\"decreased\":{},\"increased\":{\"6699b187-d17c-4783-bc8c-3263690d4aac\":{\"amount\":1,\"servicePlanAssignmentId\":21620}},\"serviceName\":\"sample-saas-app-i337243\",\"servicePlanName\":\"test\",\"servicePlanUniqueIdentifier\":\"sample-saas-app-i337243-test\",\"isEntitlementUnlimited\":false,

```

```

    \"assignmentLevel\": \"SUBACCOUNT\"}, \"jobscheduler-standard\": {\"decreased\": {},\n    \"increased\": {\"6699b187-d17c-4783-bc8c-3263690d4aac\": {\"amount\":\n        5,\n        \"servicePlanAssignmentId\": 21621}},\n    \"serviceName\": \"jobscheduler\", \"servicePlanName\": \"standard\", \"servicePlanUniqueIdentifier\": \"jobscheduler-\n        standard\", \"isEntitlementUnlimited\": false, \"assignmentLevel\": \"SUBACCOUNT\"},\n    \"cis-central\": {\"decreased\": {},\n        \"increased\": {\"6699b187-d17c-4783-\n            bc8c-3263690d4aac\": {\"amount\": 5,\n            \"servicePlanAssignmentId\": 21622}},\n        \"serviceName\": \"cis\", \"servicePlanName\": \"central\", \"servicePlanUniqueIdentifier\": \"cis-central\", \"isEntitlementUnlimited\": false,\n        \"assignmentLevel\": \"SUBACCOUNT\"}}},\n    \"subaccountIdToRegion\": {\"6699b187-d17c-4783-bc8c-3263690d4aac\":\n        \"us10-staging\"}\n    },\n    \"globalAccountGUID\": \"a7410cf4-84a5-46ac-9c15-52e34d283eb2\", \"entityId\": \"a7410cf4-84a5-46ac-9c15-52e34d283eb2\", \"entityType\": \"GlobalAccount\", \"eventOrigin\": \"entitlements-service\", \"eventType\": \"SubaccountEntitlements_Update\"\n}
]
}
}

```

## Related Information

[Manage Authentication and Authorization Process for Cloud Management APIs \[page 881\]](#)

[Using the Events Service APIs \[page 890\]](#)

[Get Event Types \[page 903\]](#)

### 5.1.3.4.2 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 [Cloud Management - Service Plans \[page 885\]](#).

## 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	Category to which the event type belongs.  Possible values: <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.

Parameter Name	Parameter Type	Description
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; border-radius: 5px;"><p> Example</p><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, AccountDirectoryTenant_Creation, AccountDirectoryTenant_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

[Manage Authentication and Authorization Process for Cloud Management APIs \[page 881\]](#)

[Using the Events Service APIs \[page 890\]](#)

[Get Events \[page 891\]](#)

### 5.1.3.5 Error Response Format

Describes the response format for the SAP Cloud Platform Cloud Management service errors.

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
11XXX	General request errors
12XXX	General operations failures
2XXXX	Accounts service errors
3XXXX	Entitlements service errors
40XXX	Tenants operations failures
41XXX	Environment instances operations failures
42XXX	Other provisioning service errors
5XXXX	Service manager proxy errors
6XXXX	Order processing service errors
7XXXX	CLI backend errors
8XXXX	Asynchronous jobs failures
9XXXX	SaaS manager service errors
10XXXX	Events service errors

## 5.1.3.6 Asynchronous Jobs

Describes the asynchronous calls that some SAP Cloud Platform Cloud Management service functionality supports.

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>"  
}
```

### i Note

The response includes the `status` (always) and `description` (optional) fields.

For example, you can trigger a brand new job.

## 5.1.3.7 Rate Limiting

Describes how all API requests to the SAP Cloud Platform Cloud Management service 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"  
    ]  
  }  
}
```

```
        }
    ]
}
}
```

#### i 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).

## 5.1.4 Manage Authentication and Authorization Using APIs

The REST services of the SAP Cloud Platform Authorization and Trust Management service (XSUAA) provide APIs that enable you to manage entities, such as roles, shadow users, and access tokens in SAP Cloud Platform, subaccounts.

To access the API, you need an OAuth 2.0 client. For more information about how to enable the `apiaccess` plan.

The following table provides an overview of the APIs. For more information about the APIs and their API endpoints, see [SAP API Business Hub](#) in the related links.

#### i Note

For subaccounts based on the Neo environment, use the Neo APIs. For more information, see [Using Platform APIs](#) in the documentation for SAP Cloud Platform, Neo environment in the related links.

APIs of SAP Cloud Platform SAP Cloud Platform Authorization and Trust Management Service

API	Description
Authorizations	Provides functions to administrate the SAP Cloud Platform Authorization and Trust Management service of SAP Cloud Platform, Cloud Foundry environment. Manage service instances of the SAP Cloud Platform 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 SAP Cloud Platform, Cloud Foundry environment.
Security Settings	Provides functions to manage the security settings of an account in SAP Cloud Platform, Cloud Foundry environment. Use this API to manage access token validity and signing keys.

API	Description
User Management System for Cross-domain Identity Management (SCIM)	Provides functions to administrate the SAP Cloud Platform Authorization and Trust Management service of SAP Cloud Platform, 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.

## Related Information

[Get API Access \[page 912\]](#)

[APIs of SAP Cloud Platform Authorization and Trust Management service on SAP API Business Hub!\[\]\(dcacec79edf3843e37349f43b44c9bc7\_img.jpg\)](#)

[Using Platform APIs](#)

### 5.1.4.1 Get API Access

To enable programmatic access to the XS user authentication and authorization (UAA) service in your subaccount of the Cloud Foundry environment, create an XSUAA service instance under the `apiaccess` plan.

## Prerequisites

- You have the role *User & Role Administrator* for the subaccount where you want to enable API access. This role ensures that you have the required scopes:
  - `xs_authorization.read`
  - `xs_authorization.write`
  - `xs_idp.read`
  - `xs_idp.write`
  - `xs_user.read`
  - `xs_user.write`
- You have an org and a space where you can create a service instance.

## Context

One use case for this scenario is to use your own identity management system to integrate it with SAP Cloud Platform. The API is a RESTful API that includes access to authorization, user, group, and identity provider interfaces. The user, group, and identity provider interfaces use System for Cross-domain Identity Management (SCIM) protocol.

For more information about the available APIs, see <https://api.sap.com/package/authtrustmgmnt> on SAP API Business Hub.

## Procedure

1. Navigate to the space of your subaccount.

Enter the following command:

```
cf target -o <org_name> -s <space_name>
```

For example:

```
cf target -o my-org -s DEV
```

2. In your subaccount, create a service instance of the UAA with the `apiaccess` plan.

Enter the following command:

```
cf create-service xsuaa apiaccess <access_name>
```

For example:

```
cf create-service xsuaa apiaccess my-access
```

This command creates an entry for the OAuth client in the database of the cloud controller.

3. Create a service key.

Enter the following command:

```
cf create-service-key <access_name> <key_name>
```

For example:

```
cf create-service-key my-access my-access-key
```

The system writes the credentials for the OAuth client to the service key.

4. Get the credentials for the OAuth client.

Enter the following command:

```
cf service-key <access_name> <key_name>
```

For example:

```
cf service-key my-access my-access-key
```

The system answers similar to the following:

### ↳ Output Code

```
Getting key my-access-key for service instance my-access as my-user...
{
  "apiurl": "https://api.authentication.eu10.hana.ondemand.com",
  "clientid": "aa-bb-cccc11c1-d222-333e-44f4-g5g55ggg555g!a6666",
  "clientsecret": "aA1B2CcCCC3dDd+ee444fFF5ggG=",
  "identityzone": "my-subdomain",
  "identityzoneid": "a11aaaa1-22b2-33c3-dd44-5555f5555f55",
  "sburl": "https://internal-xsuaa.authentication.eu10.hana.ondemand.com",
  "tenantid": "a11aaaa1-22b2-33c3-dd44-5555f5555f55",
```

```

    "tenantmode": "dedicated",
    "uaadomain": "authentication.eu10.hana.ondemand.com",
    "url": "https://my-subdomain.authentication.eu10.hana.ondemand.com",
    "verificationkey": "-----BEGIN PUBLIC KEY-----\n-----END PUBLIC KEY-----",
    "xsappname": "aa-bbbb11b1-c222-333d-44e4-f5f55fff555f!a6666"
}

```

With the URL, client ID, and client secret, you can get the access token. With the access token and the apiurl, you have access to the API.

- Configure or program your application to get a token from the OAuth client.

In your call to the OAuth client, you send the client ID and client secret, separated with a colon (:) and base64 encoded to the URL from the service key.

#### ↳ Sample Code

```

curl --request POST \
  --url https://my-subdomain.authentication.eu10.hana.ondemand.com/oauth/token \
  --header 'Accept: application/json' \
  --header 'Authorization: Basic eW91ckNsaWVudE1EOnlvdXJDbG1lbRTZWNyZXQ' \
  --header 'cache-control: no-cache' \
  --data 'grant_type=client_credentials&response_type=token'

```

The client returns a token.

#### ↳ Output Code

```

{
  "access_token": "eyJhbGciOiJSUzI1N...",
  "token_type": "bearer",
  "expires_in": 43199,
  "scope": "xs_user.write uaa.resource xs_authorization.read\nxs_idp.write xs_user.read xs_idp.read xs_authorization.write",
  "jti": "be340353ac694b4cb504c6823f938647"
}

```

- Use the value of the parameter `access_token` to make calls to the various API endpoints. For an example, see [Call an API \[page 915\]](#).

## Related Information

<https://docs.cloudfoundry.org/devguide/services/service-keys.html>

[SAP Note 2760424 - API Access to xsuaa Configuration Data](#)

<https://api.sap.com>

## 5.1.4.2 Call an API

Use the different endpoints of the Authorization and Trust Management service APIs to manage users roles and other authorization configurations.

### Prerequisites

To call an API of the Authorization and Trust Management Service you need to obtain an access token. See [Get API Access \[page 912\]](#) to learn how to obtain an access token. See the following example how to call an endpoint with a curl request.

#### → Tip

On the API Business Hub, a sandbox system is provided for you to make example calls to the different APIs and their endpoints.

### Procedure

Call the roles endpoint of the authorization API.

#### ↳ Sample Code

In this example, we request the list of roles of an XSUAA configuration.

```
curl --request GET \
  --url https://api.authentication.eu10.hana.ondemand.com/sap/rest/
authorization/v2/roles \
  --header 'Accept: application/json' \
  --header 'Authorization: bearer eyJhbGciOiJSUzI1N...' \
  --header 'Content-Type: application/x-www-form-urlencoded' \
  --header 'cache-control: no-cache'
```

### Results

The client returns the list of roles.

#### ↳ Output Code

```
[  
  {  
    "roleTemplateName": "Viewer",  
    "roleTemplateAppId": "myapp!t1111",  
    "name": "Viewer",  
    "attributeList": [],  
    "roleCapabilityIDList": [],  
    "description": "Default instance",  
  },  
  {  
    "roleTemplateName": "Administrator",  
    "roleTemplateAppId": "myapp!t1112",  
    "name": "Administrator",  
    "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 API Business Hub](#) 

## 5.2 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 in the Cockpit \[page 754\]](#)
- [Account Administration Using the CLI for SAP Cloud Platform \(sapcp CLI\) \[Feature Set B\] \[page 850\]](#)
- [Account Administration Using APIs \(Beta\) \[Feature Set B\] \[page 880\]](#)
- [Org Administration Using the Cockpit \[page 916\]](#)
- [Org Administration Using the Cloud Foundry CLI \[page 930\]](#)
- [Application Operations in the Cloud Foundry Environment \[page 979\]](#)
- [Make Your Custom Identity Provider Visible in the Login Screen \[page 987\]](#)
- [Manage Authentication and Authorization Using APIs \[page 911\]](#)
- [Audit Logging in the Cloud Foundry Environment \[page 988\]](#)

### 5.2.1 Org Administration Using the Cockpit

In the Cloud Foundry environment, manage orgs, spaces and space quota plans using the SAP Cloud Platform 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 \[page 918\]](#)
- [Managing Spaces \[page 922\]](#)
- [Managing Space Quota Plans \[page 927\]](#)

### 5.2.1.1 Navigate to Orgs and Spaces

To administer your Cloud Foundry environment, navigate to orgs, and spaces in the SAP Cloud Platform cockpit.

#### Prerequisites

- Sign up for an enterprise or a trial account and receive your logon data.  
For more information, see [Get a Free Trial \[page 7\]](#) or [Purchase a Customer Account \[page 8\]](#).
- Create the org or space to which you want to navigate.  
For more information, see [Create Orgs \[page 918\]](#) and [Create Spaces \[page 922\]](#).

#### 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="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><p><b>i 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>1. 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>2. In the navigation area, choose  <a href="#">Cloud Foundry</a> <a href="#">Spaces</a> </li></ol>

## Related Information

[Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)

[Navigate to Global Accounts and Subaccounts \[Feature Set B\] \[page 757\]](#)

### 5.2.1.2 Managing Orgs

Learn how to create a Cloud Foundry org, add members to it and later delete it if needed.

- [Create Orgs \[page 918\]](#)
- [Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#)
- [Delete Orgs \[page 921\]](#)

#### i 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 Cloud Platform cockpit. For example, renaming an org using the cf CLI won't be reflected in the cockpit.

#### 5.2.1.2.1 Create Orgs

Once a subaccount is created in the Cloud Foundry environment, you must create an org in order to use it.

## Procedure

1. Navigate to your Cloud Foundry subaccount.
2. In the subaccount *Overview* page, choose *Enable Cloud Foundry*.
3. Enter a name for your org and choose *Create*.

#### i Note

Once you've created the org, you cannot change it afterwards in the SAP Cloud Platform cockpit. If you rename the org using the Cloud Foundry CLI (cf CLI), the new org name won't be reflected in the cockpit.

## Related Information

[Navigate to Orgs and Spaces \[page 917\]](#)

[Create a Subaccount \[Feature Set A\] \[page 762\]](#)

[Create a Subaccount \[Feature Set B\] \[page 764\]](#)

## 5.2.1.2.2 Add Org Members Using the Cockpit [Feature Set A]

You can add org members and assign roles to them at the subaccount level in the cockpit.

### Prerequisites

You have the Org Manager role for the org in question.

#### i Note

You automatically have the Org Manager role in a subaccount that you created.

### Context

#### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

### Procedure

1. Choose the subaccount that contains the org to which you'd like to add members.
2. In the navigation area, choose  [Cloud Foundry](#)  [Org Members](#).  
All members currently assigned to the org are shown in a list.
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses.  
You can use commas, spaces, semicolons, or line breaks to separate members.
5. Choose the [Origin](#).  
If you want to use a custom user base, choose [Other](#) for [Origin](#) and then enter the corresponding Identity Authentication tenant name. For more information, see [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#).
6. Select the roles for the members and save your changes.

## Next Steps

- To select or unselect roles for a member, choose  ([Edit](#)). The changes you make to the roles of a member take effect immediately.
- To remove all the roles of a member, choose  ([Delete](#)). This removes the member from the organization.

 **i Note**

You can only remove org members who are no longer assigned to any space in the org.

## Related Information

[Log On to Your Global Account \[Feature Set A\] \[page 754\]](#)

[Navigate to Orgs and Spaces \[page 917\]](#)

### 5.2.1.2.3 Add Org Members Using the Cockpit [Feature Set B]

You can add org members and assign roles to them at the subaccount level in the cockpit.

## Prerequisites

You have the Org Manager role for the org in question.

 **i Note**

You automatically have the Org Manager role in a subaccount that you created.

## Context

 **i Note**

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

## Procedure

1. Choose the subaccount that contains the org to which you'd like to add members.
2. In the navigation area, choose  [Cloud Foundry](#) > [Org Members](#).  
All members currently assigned to the org are shown in a list.
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses.  
You can use commas, spaces, semicolons, or line breaks to separate members.
5. Select the roles for the members and save your changes.

## Next Steps

- To select or unselect roles for a member, choose  ([Edit](#)). The changes you make to the roles of a member take effect immediately.
- To remove all the roles of a member, choose  ([Delete](#)). This removes the member from the organization.

### Note

You can only remove org members who are no longer assigned to any space in the org.

## Related Information

[Log On to Your Global Account \[Feature Set A\] \[page 754\]](#)

[Navigate to Orgs and Spaces \[page 917\]](#)

### 5.2.1.2.4 Delete Orgs

You can delete a Cloud Foundry org from a subaccount using the 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 CLI for SAP Cloud Platform (sapcp 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 \[page 917\]](#)

### 5.2.1.3 Managing Spaces

Learn how to create and delete Cloud Foundry spaces, as well as how to add members to a space.

- [Create Spaces \[page 922\]](#)
- [Add Space Members Using the Cockpit \[Feature Set A\] \[page 923\]](#)
- [Delete Spaces \[page 927\]](#)

## Related Information

[Managing Space Quota Plans \[page 927\]](#)

### 5.2.1.3.1 Create Spaces

Create spaces in your Cloud Foundry organization using the SAP Cloud Platform cockpit.

## Prerequisites

You have a Cloud Foundry organization in your Cloud Foundry subaccount, and you have the Org Manager role in that organization.

See [Create Orgs \[page 918\]](#).

## Procedure

1. Navigate to the subaccount that contains the Cloud Foundry organization in which you'd like to create a space.
2. On the subaccount *Overview* page, you have a *Spaces* table. Choose *Create Space* from the top right hand-side corner of that table.
3. Enter a space name and choose the permissions you'd like to assign to your ID, then choose *Save*.

## Next Steps

To assign quota to spaces, see [Change Space Quota Plans \[page 929\]](#).

You can change the name of a space by going to the  *Cloud Foundry*  *Spaces* page from the left hand-side navigation and choosing  (Edit) on the tile of that space. You can also create additional spaces from there.

## Related Information

[Create Orgs \[page 918\]](#)

[Create Spaces Using the Cloud Foundry Command Line Interface \[page 965\]](#)

### 5.2.1.3.2 Add Space Members Using the Cockpit [Feature Set A]

You can add space members and assign roles to them at the space level in the 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 be members of the org. For more details, see the *Context* section.
- You have the e-mail addresses of the users that you want to add.

### i Note

You must add e-mail addresses of registered members who have an S-user or a P-user (normally used for trial accounts). Administrators can request S-user IDs on the SAP ONE Support Launchpad [User Management](#) application: [1271482](#).

If users don't have a registered S-user ID, they can register for a P-user on [sap.com](#).

## Context

### i Note

The content in this section is only relevant for cloud management tools feature set A. For more information, see [Cloud Management Tools - Feature Set Overview](#).

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

1. Navigate to the space to which you'd like to add members. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).
2. In the navigation area, choose  . All members currently assigned to the space are shown in a list.
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses.  
You can use commas, spaces, semicolons, or line breaks to separate members.
5. Choose the [Origin](#).

If you want to use a custom user base, choose [Other](#) for [Origin](#) and then enter the corresponding Identity Authentication tenant name. For more information, see [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#).

6. Select the roles for the users and save your changes.

## Next Steps

You also have the following options:

- To select or unselect roles for a member, choose  ([Edit](#)). The changes you make to the roles of a member take effect immediately.
- To remove all the roles of a member, choose . This removes the member from the space.

## Related Information

[Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#)

### 5.2.1.3.3 Add Space Members Using the Cockpit [Feature Set B]

You can add space members and assign roles to them at the space level in the 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 be members of the org. For more details, see the [Context](#) section.
- You have the e-mail addresses of the users that you want to add.

#### Note

You must add e-mail addresses of registered members who have an S-user or a P-user (normally used for trial accounts). Administrators can request S-user IDs on the SAP ONE Support Launchpad [User Management](#) application: [1271482](#).

If users don't have a registered S-user ID, they can register for a P-user on [sap.com](#).

## Context

#### Note

The content in this section is only relevant for cloud management tools feature set B. For more information, see [Cloud Management Tools - Feature Set Overview](#).

Space members are org 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

1. Navigate to the space to which you'd like to add members. For more information, see [Navigate to Orgs and Spaces \[page 917\]](#).
2. In the navigation area, choose  . All members currently assigned to the space are shown in a list.
3. Choose [Add Members](#).
4. Enter one or more e-mail addresses.  
You can use commas, spaces, semicolons, or line breaks to separate members.
5. Select the roles for the users and save your changes.

## Next Steps

You also have the following options:

- To select or unselect roles for a member, choose  ([Edit](#)). The changes you make to the roles of a member take effect immediately.
- To remove all the roles of a member, choose . This removes the member from the space.

## Related Information

[Add Org Members Using the Cockpit \[Feature Set A\] \[page 919\]](#)

## 5.2.1.3.4 Delete Spaces

Delete spaces in your Cloud Foundry org using the SAP Cloud Platform 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 <https://docs.cloudfoundry.org/concepts/roles.html>.
- 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   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 \[page 922\]](#)

## 5.2.1.4 Managing Space Quota Plans

Learn how to create, assign and change space quota plans in the Cloud Foundry environment.

- [Create Space Quota Plans \[page 928\]](#)
- [Assign Quota Plans to Spaces \[page 929\]](#)
- [Change Space Quota Plans \[page 929\]](#)

### Related Information

[Managing Spaces \[page 922\]](#)

## 5.2.1.4.1 Create Space Quota Plans

You can use the cockpit to create space quota plans.

### Prerequisites

You have the Org Manager role for the org in which you want to create a space quota plan.

### 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*  *Quota Plans*.
3. Choose *New Plan*.
4. Enter a name for your quota plan and add limits for the following quotas:
  - Memory (MB): Total amount of memory a space can have
  - (Optional) Routes: Total number of routes
  - (Optional) Services: Total number of service instances
  - (Optional) Instance memory: Maximum amount of memory an application instance can have (-1 represents an unlimited amount)
  - (Optional) App Instances: Total number of application instances
  - (Optional) Allow paid services: Select if you'd like to allow the provisioning of instances of paid service plans

#### Note

The org quota limit is applicable for a resource if you don't enter a space quota limit. If the space quota limit for a resource exceeds the org quota limit, the org quota limit applies.

5. Save your changes.

### Related Information

[Entitlements and Quotas](#)

## 5.2.1.4.2 Assign Quota Plans to Spaces

You can use the SAP Cloud Platform cockpit to assign quota plans to spaces.

### Prerequisites

- You have the Org Manager role for the org in which you want to create a space quota plan.
- You have at least one space quota plan. For more information, see [Create Space Quota Plans \[page 928\]](#).

### 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](#) .
3. In the [Plan Assignment](#) section, select a quota plan for your spaces.

### Related Information

[Entitlements and Quotas](#)

## 5.2.1.4.3 Change Space Quota Plans

Manage space quota plans in the Cloud Foundry environment using the SAP Cloud Platform cockpit.

### Prerequisites

You have the Org Manager role for the org in which you want to change space quota plans.

### Procedure

1. Choose the subaccount that contains the spaces for which you'd like to change the quota.
2. In the navigation menu, choose  [Cloud Foundry](#) .
3. Choose  (Edit) for an existing quota plan.

4. Adjust the quotas in the text fields as needed.

For more information about the different values, see <https://docs.cloudfoundry.org/adminguide/quota-plans.html#space>.

5. Save your changes.

## Related Information

[Change Space Quota Plans Using the Command Line Interface \[page 968\]](#)

[Create Space Quota Plans \[page 928\]](#)

[Assign Quota Plans to Spaces \[page 929\]](#)

[Change Space Quota Plans \[page 929\]](#)

[Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#)

[Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 968\]](#)

[Change Space Quota Plans Using the Command Line Interface \[page 968\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 781\]](#)

[Entitlements and Quotas](#)

## 5.2.2 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 \[page 931\]](#)
- [CF CLI: Plug-ins \[page 936\]](#)
- [Create Spaces Using the Cloud Foundry Command Line Interface \[page 965\]](#)
- [Add Organization Members Using the Cloud Foundry Command Line Interface \[page 964\]](#)
- [Add Space Members Using the Cloud Foundry Command Line Interface \[page 966\]](#)
- [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#)
- [Assign Quota Plans to Spaces Using the Cloud Foundry Command Line Interface \[page 968\]](#)
- [Change Space Quota Plans Using the Command Line Interface \[page 968\]](#)

### i 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 Cloud Platform cockpit. For example, renaming an org using the cf CLI won't be reflected in the cockpit. In addition, you must delete an org using only the cockpit or the CLI for SAP Cloud Platform (sapcp CLI).

## 5.2.2.1 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 \[page 931\]](#)

[Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#)

[CF CLI: Plug-ins \[page 936\]](#)

### 5.2.2.1.1 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>.

## 5.2.2.1.2 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 \[Feature Set A\] \[page 762\]](#) and [Create a Subaccount \[Feature Set B\] \[page 764\]](#).
- (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 \[Feature Set A\] \[page 762\]](#).

#### i 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 \[page 931\]](#).

### Procedure

1. Open a command line.
2. Set the target API endpoint to the cloud controller.

For more information, see [Regions and API Endpoints Available for the Cloud Foundry Environment](#).

#### i Note

There is no specific endpoint for trial accounts. Both enterprise and trial accounts use the same API endpoints.

3. Log on to the Cloud Foundry environment:

```
cf login
```

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>`.

### 5.2.2.1.3 Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface

Learn how to use two 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 \[Feature Set A\] \[page 762\]](#).
- You've downloaded and installed the cf CLI. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- Your administrator has configured your Cloud Foundry environment to use a custom identity provider. For more information, see [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#).

#### ! Restriction

Trial accounts don't support the use of a custom identity provider.

#### Context

The Cloud Foundry Command Line Interface provides two options to log on using a custom IdP. One scenario is recommended for an interactive logon because it's necessary to switch from the CLI to the browser in order to log on, the other scenario is focused 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	Instructions
You have the possibility to open a browser during the logon process.	Follow these instructions: <a href="#">Log On Manually With a Custom Identity Provider [page 934]</a> .
The logon process is automated, for example with a script or there's no possibility to open a browser during logon.	Follow these instructions: <a href="#">Set Up Automated Logon With a Custom Identity Provider [page 935]</a>

## Related Information

[Make Your Custom Identity Provider Visible in the Login Screen \[page 987\]](#)

### 5.2.2.1.3.1 Log On Manually With a Custom Identity Provider

To log on to Cloud Foundry, using a custom identity provider (IdP), use the single sign-on option of the CF CLI. During logon, the system provides you with a URL to your custom IdP for you to enter in a web browser. If you're already logged on to your custom IdP, the system provides a temporary authentication code to complete your logon. If you don't have an active session with your custom IdP, you need to log on to receive the temporary authentication code.

#### Prerequisites

- You've created at least one subaccount and enabled the Cloud Foundry environment in this subaccount.  
For more information, see [Create a Subaccount \[Feature Set A\] \[page 762\]](#).
- You've downloaded and installed the cf CLI. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- You've configured the login screen to display your custom identity provider, see [Make Your Custom Identity Provider Visible in the Login Screen \[page 987\]](#).
- Your administrator has configured your Cloud Foundry environment to use a custom identity provider.  
For more information, see [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#).

#### ! Restriction

Trial accounts don't support the use of a custom identity provider.

#### 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.

Depending on your setup, you can now choose between different accounts, for example the default identity provider (SAP ID service) and your custom identity provider.

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 IdP.

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

[Make Your Custom Identity Provider Visible in the Login Screen \[page 987\]](#)

### 5.2.2.1.3.2 Set Up Automated Logon With a Custom Identity Provider

To log on to Cloud Foundry, using a custom identity provider (IdP), use the `--origin` option of the CF CLI.

## Prerequisites

- You've created at least one subaccount and enabled the Cloud Foundry environment in this subaccount. For more information, see [Create a Subaccount \[Feature Set A\] \[page 762\]](#).
- You've downloaded and installed the cf CLI. For more information, see [Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#).
- Your administrator has configured your Cloud Foundry environment to use a custom identity provider. For more information, see [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#).
- Users are maintained in the Identity Authentication Service and not in the custom identity provider.

#### ! Restriction

Trial accounts don't support the use of a custom identity provider.

## Context

We recommend this method of logging on if you want to use an automated script and can't use the substep of opening 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](#).
- Find the <origin> value for the user that you want to use in the Cloud Cockpit.
  1. Navigate to the subaccount of your user.
  2. Choose [Members](#).
  3. Find the <origin> value of your user in the table.
- If two-factor authentication is enabled, you have to concatenate your password and the passcode.

### 5.2.2.1.4 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 \[page 938\]](#)

#### 5.2.2.1.4.1 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.

##### i Note

Before using the extended commands in the Cloud Foundry environment, you need to install the MTA plug-in in the Cloud Foundry environment.

## Related Information

[Multitarget Applications in the Cloud Foundry Environment \[page 286\]](#)

### 5.2.2.1.4.1.1 Install the MultiApps Plug-in in the Cloud Foundry Environment

The MultiApps plug-in (formerly known as the MTA plug-in) 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 6.20 or higher.

#### 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 plug-in and try to install a new one, you'll 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 plug-in, proceed as follows:

```
cf install-plugin multiapps -f
```

#### i Note

Alternatively, if you want to install a specific version of the plug-in, proceed as follows:

1. Download the [preferred version](#) of the plug-in that is compatible with your operating system.
2. Untar or unzip the downloaded archive if required.
3. To install the plug-in, enter the following command:
  - Mac and Linux: `cf install-plugin<path to the downloaded plug-in> -f`

- Windows : `cf install-plugin<path to the downloaded plug-in> -f`
- Verify that the plug-in has been installed successfully by entering `cf plugins`.

You see a list of plug-ins that now includes the MTA plug-in for the Cloud Foundry command line interface. The output also displays commands that are specific to this plug-in.

## Related Information

<https://tools.hana.ondemand.com/#cloud>

[Download and Install the Cloud Foundry Command Line Interface \[page 931\]](#)

## 5.2.2.1.4.1.2 Multitarget Application Commands for the Cloud Foundry Environment

A list of additional commands to install archives and deploy Multitarget Applications (MTA) to the Cloud Foundry environment.

### i Note

The expiration time for all Cloud Foundry operations is 3 days. If an operation is still active when time limit is reached, it is automatically aborted.

Commands for the Cloud Foundry Environment Overview

Command	Alias	Description
<code>deploy</code>		Deploy a new Multitarget Application (MTA) or synchronize changes to an existing MTA
<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

# deploy

Deploy a new Multitarget Application (MTA) or synchronize changes to an existing MTA.

## Usage

Deploy a new Multitarget Application or synchronize changes to an existing one:

```
cf deploy <MTA_ARCHIVE>|<DIRECTORY_PATH> [-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] [--do-not-fail-on-missing-  
permissions]
```

Interact with an active MTA deploy operation, for example, by performing an action:

```
cf deploy [-i <OPERATION_ID>] [-a <ACTION>]
```

## Arguments

The MTA 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> .
<DIRECTORY_PATH>	The 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> .

## Options

Command Options Overview

Option	Description
<code>-e</code> <code>&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 deployment 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

Option	Description
-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
--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. <div style="border-left: 3px solid #0070C0; padding-left: 10px; margin-top: 10px;"> <b>i Note</b>            It can be used multiple times.         </div>
--all-modules	Deploy all modules. <div style="border-left: 3px solid #0070C0; padding-left: 10px; margin-top: 10px;"> <b>i Note</b>            Any -m options are ignored.         </div>
-r <resource name>	Deploy only the resource with the specified name <div style="border-left: 3px solid #0070C0; padding-left: 10px; margin-top: 10px;"> <b>i Note</b>            It can be used multiple times.         </div>

Option	Description
--all-resources	<p>Deploy all resources.</p> <p><b>i Note</b></p> <p>Any <code>-r</code> options are ignored.</p>
--verify-archive-signature	<p>Check the archive signature by verifying that:</p> <ul style="list-style-type: none"> <li>• The archive is correctly signed with a Symantec certificate.</li> <li>• All files are signed correctly.</li> <li>• The certificate is valid.</li> <li>• The certificate subject is SAP.</li> </ul> <p><b>Example</b></p> <pre>cf deploy &lt;mtar_name&gt; --verify-archive-signature</pre>
--strategy	<p>Announce to the platform a special deployment approach, for example when performing a “blue-green” deployment. See <a href="#">Blue-Green Deployment Strategy [page 352]</a>.</p> <p><b>i Note</b></p> <p>This option is currently experimental.</p>
--skip-testing-phase	<p>When using the <code>--strategy</code> option for “blue-green” deployment, you can choose to skip the phase for testing the correct operation of the application using productive data.</p> <p><b>i Note</b></p> <p>This option is currently experimental.</p>
<p><b>i Note</b></p> <ul style="list-style-type: none"> <li>• If any of the options <code>-m</code>, <code>--all-modules</code>, <code>-r</code>, <code>--all-resource</code> 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 (<code>-m</code>, <code>--all-modules</code>) are used, the modules need to contain a <code>path</code> element on an MTA module level, which points to the content of the module. In case the module has some <code>requires</code> dependency section to a resource that needs a configuration file, then the <code>requires</code> section should have a <code>path</code> parameter, which points to the configuration file.</li> <li>• If the options for the resource deploying (<code>-r</code>, <code>--all-resource</code>) are used, then any resources that have some configuration files need to contain a <code>path</code> parameter in their parameters section, which points to the configuration file.</li> <li>• The <code>path</code> element or parameter value should be relative to the MTA directory.</li> </ul> <p>An example of an MTA deployment descriptor, containing all variants of the <code>path</code> elements and parameters can be found at <a href="#">Defining Multitarget Application Deployment Descriptors for Cloud Foundry [page 294]</a>.</p>	

For more information about

## bg-deploy

Deploy a new Multitarget Application (MTA) using “blue-green” (zero-downtime) deployment.

### i 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]
```

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)).
or	

### 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 deployment service must wait for before starting the deployed application

Option	Description
<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 deploy without requiring any confirmation for aborting any conflicting processes
<code>--no-start</code>	Do not start application after deployment
<code>--use-namespaces</code>	Use namespaces in application (and service) names during application deployment
<code>--no-namespaces-for-services</code>	Do not use namespaces in service names
<code>--delete-services</code>	Re-create changed services and delete discontinued services
<code>--delete-service-keys</code>	Delete the existing service keys and apply the new ones
<code>--delete-service-brokers</code>	Delete discontinued service brokers
<code>--keep-files</code>	Keep the files used for deployment
<code>--no-restart-subscribed-apps</code>	Do not restart subscribed applications that are updated during the deployment
<code>--no-confirm</code>	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.
<code>--do-not-fail-on-missing-permissions</code>	Perform the deployment, even if required administrator permissions are missing for some operations (for example, the creation of service brokers).
<code>--abort-on-error</code>	If an operation fails, the corresponding process is automatically aborted and cannot be retried using the option <code>-a retry</code>
<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.	Deploy only the module with the specified name.
<code>--all-modules</code>	Deploy all modules.

**i Note**

It can be used multiple times.

**i Note**

Any `-m` options are ignored.

Option	Description
<code>-r &lt;resource name&gt;</code>	<p>Deploy only the resource with the specified name</p> <div style="border-left: 2px solid #0070C0; padding-left: 10px;"> <b>i Note</b>            It can be used multiple times.         </div>
<code>--all-resources</code>	<p>Deploy all resources.</p> <div style="border-left: 2px solid #0070C0; padding-left: 10px;"> <b>i Note</b>            Any <code>-r</code> options are ignored.         </div>
<code>--verify-archive-signature</code>	<p>Check the archive signature by verifying that:</p> <ul style="list-style-type: none"> <li>• The archive is correctly signed with a Symantec certificate.</li> <li>• All files are signed correctly.</li> <li>• The certificate is valid.</li> <li>• The certificate subject is SAP.</li> </ul> <div style="border-left: 2px solid #0070C0; padding-left: 10px;"> <b>❖ Example</b>  <code>cf bg-deploy &lt;mтар_name&gt; --verify-archive-signature</code> </div>

### i 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 \[page 350\]](#).

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 \[page 294\]](#).

## 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]
```

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

## 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).
--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

Option	Description
-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.

**i 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

## 5.2.2.1.4.1.2.1 (Experimental) Namespaces

Use this feature to prevent conflicts for applications deriving from the same MTA, but with different version, features, and configuration.

**i Note**

This is an improved implementation of a previous feature with the same name.

**! Restriction**

- This feature is experimental. We advise you first try it out in a non-productive environment.

- There are specific deployment descriptor modifications that need to be employed. The modifications are described in this section.
- The namespace can only be passed as an argument to the deployment call, and cannot be modelled directly in the descriptor.

## 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 Plug-in in the Cloud Foundry Environment \[page 937\]](#) to update it.

## Feature Description

Using namespaces allows you to deploy MTAs with the same ID multiple times in a single space. That is, you can have multiple versions of an MTA operating at the same time.

### Sample Code

```
...
modules:
  - name: app
    type: java.tomee
    path: webapp.war
    parameters:
      routes:
        - route: ${app-name}.${default-domain}
...
resources:
  - name: service
...

```

In the example above:

- The `route` parameter value contains the  `${app-name}`  parameter to ensure the route is unique among the multiple deployments. The module and resource names are automatically changed during deployment, that is, they receive a prefix with the provided namespace. You may want to do this modification as this ensures the application routes assigned are unique.
- No changes are required to the `resources` section of the descriptor. The service names receive a prefix automatically.

## Deployment

To deploy an MTA using a namespace, use the following command:

```
cf deploy demo.mtar --namespace foo
```

After deployment, the application in the archive is created as `foo-app` and the service as `foo-service`.

## **Skipping the namespace prefix**

If you want to omit the namespace applied to a particular module or resource, you have to set the parameter `apply-namespace` appropriately in the descriptor:

### Sample Code

```
...
modules:
- name: app1
  type: java.tomee
  path: webapp.war
  parameters:
    apply-namespace: false
...
resources:
- name: service1 #no changes required to resources
  parameters:
    apply-namespace: false
...
```

In the example above the `apply-namespace` parameter is set to `false`, which overrides the default value `true`. Thus, no namespace prefix is added to the application name. This ensures that during deployment with namespace, `app1` and `service1` are created as usual, and operate in the general MTA behaviour for duplicated applications and services. The `apply-namespace` parameter can be applied to resources as well.

## **Administration of namespace MTAs**

- To view deployed MTAs, use the `cf mtas` command:

### Sample Code

```
$ cf mtas
Getting multi-target apps in org <your organization> / space <your space>
as user...
OK
mta id          version  namespace
demo           1.0.0    bar
demo           1.0.0    foo
demo           1.0.0
```

- To view detailed information about an MTA with namespace, use the command:

```
cf mta demo --namespace foo
```

- To undeploy an MTA using a namespace, you need to pass the same parameter again:

```
cf undeploy demo --namespace foo
```

## **Related Information**

[Deploying an MTA in a Namespace](#) 

## 5.2.2.1.4.1.2.2 Application Versions and Deployment Consistency

The deployment service follows the rules specified in the Semantic Versioning Specification (Semver) when comparing the versions of a deployed MTA with the version that is to be deployed. If an MTA 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 deployment 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.

## 5.2.2.1.4.2 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 Cloud Platform cockpit or the extended Cloud Foundry commands in the command line interface.

### 5.2.2.1.4.2.1 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]>`

**i 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 \[page 931\]](#)

## 5.2.2.1.4.2.2 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>

## 5.2.2.1.4.3 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 \[page 955\]](#)

[Extended Cloud Foundry Commands of Custom Domains \[page 957\]](#)

[Configuring Custom Domains](#)

### 5.2.2.1.4.3.1 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 \[page 931\]](#).

#### 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 Cloud Platform 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
```

### **i** 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 \[page 931\]](#)

[Configuring Custom Domains](#)

## 5.2.2.1.4.3.2 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]	- skip - ssl- vali dati on - verb ose - forc e	Generate a new private and public key pair. Do not attempt to validate SSL certificate. Show verbose information. Force key creation without confirmation.
cf custom-domain-get-csr	cdgc	cf custom-domain-get-csr KEY [FILE] [options]	- skip - ssl- vali dati on - verb ose	Download the certificate signing request corresponding to a new key. Do not attempt to validate SSL certificate. Show verbose information.

Command	Alias	Usage	Options	Description
cf custom-domain-upload-certificate-chain	cducc	cf custom-domain-upload-certificate-chain KEY FILE [options]	- skip - ssl- vali dati on - verb ose - forc e	Upload a certificate chain for a given key.  Do not attempt to validate SSL certificate.  Show verbose information.  Force uploading the certificate chain without confirmation.
cf custom-domain-activate	cda	cf custom-domain-activate KEY DOMAIN [...] [options]	- skip - ssl- vali dati on - verb ose	Activate the server certificate for specified domains.  Do not attempt to validate SSL certificate.  Show verbose information.

Command	Alias	Usage	Options	Description
cf custom-domain-list	cdl	cf custom-domain-list	<ul style="list-style-type: none"> <li>- skip Do not attempt to validate SSL certificate.</li> </ul> <hr/> <ul style="list-style-type: none"> <li>- verbose Show verbose information.</li> </ul>	List domains and their configuration status.
cf custom-domain-show-certificates	cdsc	cf custom-domain-show-certificates KEY [FILE] [options]	<ul style="list-style-type: none"> <li>- dump Dump the certificates in PEM format.</li> </ul> <hr/> <ul style="list-style-type: none"> <li>- skip Do not attempt to validate SSL certificate.</li> </ul> <hr/> <ul style="list-style-type: none"> <li>- verbose Show verbose information.</li> </ul>	Show certificates for a specified key.

Command	Alias	Usage	Options	Description
cf custom-domain-enable-client-authentication	cdeca	cf custom-domain-enable-client-authentication [FILE DOMAIN [DOMAIN ...] [options]]	- skip - ssl-validation - datation - verbbose	Do not attempt to validate SSL certificate. Show verbose information.
cf custom-domain-disable-client-authentication	ccddca	cf custom-domain-disable-client-authentication DOMAIN [options]	- skip - ssl-validation - datation - verbbose - force	Disable client authentication for a domain. 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]	<ul style="list-style-type: none"> <li>- dump Dump the certificates in PEM format.</li> <li>- skip Do not attempt to validate SSL certificate.</li> <li>- ssl-validation Show verbose information.</li> </ul>	Show the trusted certificates for a domain.
cf custom-domain-deactivate	cdd	cf custom-domain-deactivate DOMAIN [options]	<ul style="list-style-type: none"> <li>- skip Do not attempt to validate SSL certificate.</li> <li>- ssl-validation Show verbose information.</li> <li>- force Force deactivation without confirmation.</li> </ul>	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]	<ul style="list-style-type: none"> <li>- skip Do not attempt to validate SSL certificate.</li> <li>- verbose Show verbose information.</li> <li>- force Force key deletion without confirmation.</li> </ul>	Delete the private key and its certificates.
cf custom-domain-version	cdv	cf custom-domain-version	<ul style="list-style-type: none"> <li>- skip Do not attempt to validate SSL certificate.</li> <li>- verbose Show verbose information.</li> </ul>	Show the Custom Domain plugin version.

Command	Alias	Usage	Options	Description
cf custom-domain-map-route	cdmr	cf custom-domain-map-route STANDARD_ROUTE E CUSTOM_ROUTE	- skip - ssl-validation - validate SSL certificate. - Show verbosebose information.	Map a custom domain route to a SaaS application.
cf custom-domain-unmap-route	cdur	cf custom-domain-unmap-route CUSTOM_ROUTE	- skip - ssl-validation - validate SSL certificate. - Show verbosebose information.	Unmap a custom domain route from a SaaS application.
cf custom-domain-routes	cdr	cf custom-domain-routes	- skip - ssl-validation - validate SSL certificate. - Show verbosebose information.	List all configured routes of SaaS applications with custom domains.

## Related Information

[Configuring Custom Domains](#)  
[Managing Custom Domains](#)

### 5.2.2.2 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. You can enable it in the Cloud Platform Cockpit. For more information, see [Create a Subaccount \[Feature Set A\] \[page 762\]](#) and [Create a Subaccount \[Feature Set B\] \[page 764\]](#).
- The Org Manager role for the org in question.

#### i Note

You automatically have the Org Manager role in a subaccount that you created.

- The e-mail addresses of the members that you want to add
- 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

#### Context

When you logged on with a platform user in a custom platform identity provider using the Cloud Foundry Command Line Interface, you can't use the `--origin` option. This isn't possible with the current version of the Cloud Foundry Command Line Interface (cf CLI). For more information, see [Supported Tools and Services When Using Custom Platform Identity Providers \[page 804\]](#).

#### Procedure

Enter the following string, specifying the user name, the name of the organization, and the role:

```
cf set-org-role USERNAME ORG ROLE
```

## 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
```

### 5.2.2.3 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 \[Feature Set A\] \[page 762\]](#) and [Create Orgs \[page 918\]](#).

**i 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

#### 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]
```

- (Optional) Set this space as target by executing: `cf target -o ORG -s SPACE`

**i 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 \[page 969\]](#)

### 5.2.2.4 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 e-mail addresses of the users that you want to add
- 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

#### Context

When you logged on with a platform user in a custom platform identity provider using the Cloud Foundry Command Line Interface, you can't use the `--origin` option. This isn't possible with the current version of the Cloud Foundry Command Line Interface (cf CLI). For more information, see [Supported Tools and Services When Using Custom Platform Identity Providers \[page 804\]](#).

#### 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
```

### 5.2.2.5 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

#### 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

## 5.2.2.6 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).
- A space quota plan. For more information, see [Create Space Quota Plans Using the Cloud Foundry Command Line Interface \[page 967\]](#).

### Procedure

Open a command line and enter the following string:

```
cf set-space-quota SPACE_NAME SPACE_QUOTA_NAME
```

## 5.2.2.7 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 \[page 932\]](#).

## 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]
```

### i Note

For more information about managing space quota plans, see <https://docs.cloudfoundry.org/adminguide/quota-plans.html#space>.

## Related Information

[Change Space Quota Plans \[page 929\]](#)

[Configure Entitlements and Quotas for Subaccounts \[page 781\]](#)

[Managing Entitlements and Quotas Using the Cockpit \[page 777\]](#)

### 5.2.2.8 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 \[page 931\]](#) and [Log On to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 932\]](#).

## 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 \[page 965\]](#)

### 5.2.2.9 Managing Secrets of the SAP Cloud Platform Authorization and Trust Management Service

The SAP Cloud Platform 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 Cloud Platform Authorization and Trust Management service uses the following secrets in its operation:

- Service instance secrets

## Related Information

[Service Instance Secrets \[page 970\]](#)

### 5.2.2.9.1 Service Instance Secrets

When an application consumes a service instance of SAP Cloud Platform Authorization and Trust Management service (XSUAA), the application identifies itself to the service instance with a client ID and client secret. The

client ID and client 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.

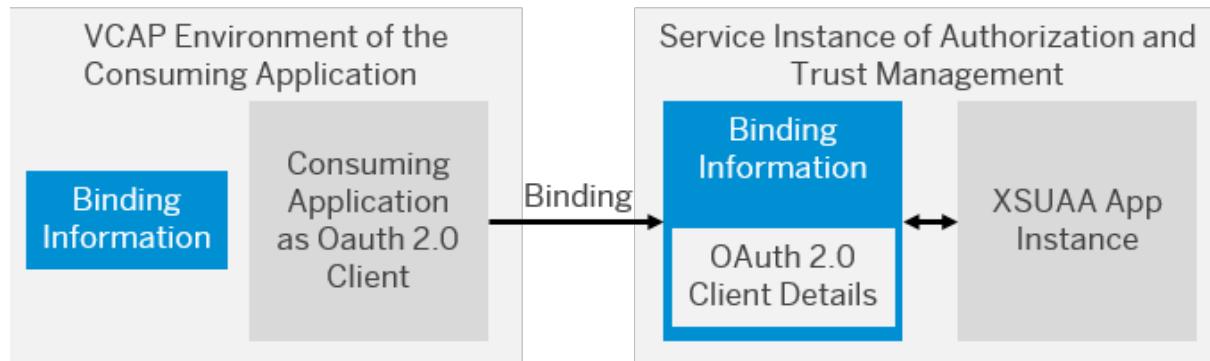
The service instance can use multiple secrets in the application plan.

- The instance secret is the default secret. The secret is the same for all bindings of the service instance. The secret remains valid as long as the instance exists.
- The binding secret must be enabled in the application security descriptor (`xs-security.json`) when you deploy the application. The secret remains valid as long as the binding or the service key exists.

#### i Note

The `apiaccess` plan only uses binding secrets. However, some old instances of the `apiaccess` plan might still use the instance secret.

The following figure illustrates the XSUAA app and its information about the OAuth 2.0 client as part of an SAP Cloud Platform Authorization and Trust Management service instance. A consuming application, functioning as an OAuth 2.0 client is bound to the SAP Cloud Platform Authorization and Trust Management service instance. The instance 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 Cloud Platform 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 secrets bindings support.

In the following example, the service instance creates a binding secret for all new bindings, but still accepts the instance secret.

#### ❖ Example

##### ↳ Sample Code

```
"oauth2-configuration": {  
    "credential-types": ["binding-secret", "instance-secret"]  
}
```

In the following example, the service instance creates a binding secret for all new bindings, but doesn't accept the instance secret.

### • Example

#### ↳ Sample Code

```
"oauth2-configuration": {  
    "credential-types": ["binding-secret"]  
}
```

## Related Information

[Rotate Binding Secrets \[page 972\]](#)

[Rotate Instance Secrets \[page 974\]](#)

[The Application Security Descriptor \[page 253\]](#)

[Migrate from Instance Secrets to Binding Secrets \[page 973\]](#)

### 5.2.2.9.2 Rotate Binding Secrets

Service instances of the SAP Cloud Platform Authorization and Trust Management service use different binding secrets for each binding. To rotate binding secrets, unbind and rebind any consuming applications.

## Prerequisites

- You've enabled support for binding secrets.  
For more information, see [Migrate from Instance Secrets to Binding Secrets \[page 973\]](#).
- If you have to replace any service keys, create the new service keys before proceeding.  
For more information, see [Creating Service Keys \[page 379\]](#).

## Procedure

1. Unbind the application that consumes your service instance.
2. Rebind the application that consumes your service instance.

## Results

When you rebind the application, the system creates a new binding secret.

## Next Steps

Delete any old service keys that you don't need anymore.

## Related Information

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

[Service Instance Secrets \[page 970\]](#)

### 5.2.2.9.3 Migrate from Instance Secrets to Binding Secrets

To simplify the management of secrets for service instances of the SAP Cloud Platform Authorization and Trust Management service, we recommend that you configure service instances to use binding secrets.

## Context

By default, service instances of the SAP Cloud Platform 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 \[page 279\]](#)

[Creating Service Keys \[page 379\]](#)

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

[Service Instance Secrets \[page 970\]](#)

## 5.2.2.9.4 Rotate Instance Secrets

A service instance of the SAP Cloud Platform Authorization and Trust Management service uses the same instance secret for all bindings. To rotate instance credentials, create a new service instance and then rebind all the applications that consume the old instance to the new instance.

## Context

### → Recommendation

We recommend that you migrate to using binding secrets instead of instance secrets. If you lose one binding secret, then you don't need to replace all bindings.

For more information, see [Migrate from Instance Secrets to Binding Secrets \[page 973\]](#).

## Procedure

1. Create a new service instance of the SAP Cloud Platform Authorization and Trust Management service.

Creating a new service instance creates a new instance secret.

2. Create any service keys for the new service instance.
3. Bind any applications that consume your old service instance to the new service instance.
4. Delete the old service instance.

## Related Information

[Create a Service Instance from the xsuaa Service \[page 278\]](#)

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

[Deleting Service Instances \[page 381\]](#)

[Service Instance Secrets \[page 970\]](#)

### 5.2.2.9.5 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. To rotate the signing keys of access token, use the Security Setting API of the SAP Cloud Platform Authorization and Trust Management service.

#### Prerequisites

- You've enabled API access to the SAP Cloud Platform Authorization and Trust Management service.  
For more information, see [Get API Access \[page 912\]](#)
- You've checked that you have space for an additional signing key.  
You can store two signing keys per service instance. Get the settings of your subaccount to see how many keys you've stored there. Delete an inactive key if you already have two.

#### Context

Access tokens are JSON web tokens (JWT). The system uses the signing key of the access token to digitally sign the JWT.

A service instance of the SAP Cloud Platform Authorization and Trust Management service has an initial signing key for its access token.

##### ⚠ Caution

When you enable the first signing key after the initial signing key, the initial signing key is immediately invalid. Clients with access tokens issued within the last 12 hours, the default lifetime of an access token, can't use their access tokens anymore. Affected clients must reauthenticate to get a new token.

We recommend that you plan for possible service interruption when you add your first signing key.

## Procedure

1. Add a new signing key for the access token to the service instance.

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
```

Specify the tenant Id of the tenant that you want to work on with the parameter `tenantid`.

In the body, include a key ID and set `changeMode` to **ADD** in the token policy settings.

For more information, check the security settings API on the [API Business Hub](#).

### ↳ Sample Code

```
{  
    "tokenPolicySettings": {  
        "keyId": "my-new-key",  
        "changeMode": "ADD"  
    }  
}
```

The service generates a new signing key.

2. Enable the new signing key.

Call the [PATCH](#) method of the same endpoint, but set `changeMode` to **UPDATE**.

```
{  
    "tokenPolicySettings": {  
        "keyId": "my-new-key",  
        "changeMode": "UPDATE"  
    }  
}
```

The service instance now signs new access tokens with the new signing key.

3. 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.

### i Note

Exception: The initial signing key is invalid as soon as you enable your first signing key.

4. 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": "my-old-key",  
        "changeMode": "DELETE"  
    }  
}
```

### i Note

You can't delete the default signing key. The default signing key is invalid as soon as you add your first signing key.

## Related Information

[Managing Secrets of the SAP Cloud Platform Authorization and Trust Management Service \[page 970\]](#)

### 5.2.2.9.6 Rotate Signing Keys of SAML Token

The SAP Cloud Platform 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 Cloud Platform 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 Cloud Platform Authorization and Trust Management service. For more information, see [Get API Access \[page 912\]](#)
- Optionally, to use your own signing key for SAML tokens, you've obtained an X.509 certificate for the SAP Cloud Platform Authorization and Trust Management service.
- You've checked that you have space for an additional signing key. You can store two signing keys per service instance. 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 Cloud Platform Authorization and Trust Management service instance and the identity provider.

## Procedure

1. Add a new signing key for SAML tokens to the service instance.

Call the [PATCH](#) method of the Security Settings API at the following endpoint:

```
https://api.authentication.<landscape>.hana.ondemand.com/sap/rest/  
authorization/v2/tenants/<subaccount_ID>/settings
```

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"  
    }  
}
```

The service generates a new signing key.

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.

#### i Note

#### « Sample Code

```
{  
    "samlConfigSettings": {  
        "keyId": "my-new-key",  
        "changeMode": "ADD",  
        "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"  
    }  
}
```

The service instance now signs new SAML tokens 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 UAA and SAP Cloud Platform Identity Authentication Service \[page 794\]](#)
- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#)

#### ⚠ Caution

Until you upload the metadata to your identity provider, the identity provider rejects SAML tokens signed by the service instance.

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

[Get API Access \[page 912\]](#)

[Managing Secrets of the SAP Cloud Platform Authorization and Trust Management Service \[page 970\]](#)

## 5.2.3 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 \[page 980\]](#)
- [Start, Stop and Restart Applications \[page 981\]](#)
- [Add or Remove Application Instances \[page 982\]](#)
- [Configuring Application URLs \[page 983\]](#)
  - [About Routes in the Cockpit \[page 984\]](#)
  - [Create Routes \[page 984\]](#)
  - [Map Routes to Applications Using the Cockpit \[page 986\]](#)
  - [Using Custom Domains \[page 987\]](#)

### 5.2.3.1 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.
5. (Optional) If you don't want to use a manifest, untick 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.

##### i Note

By default, each instance of a new app is assigned 1024 MB of memory and 512 MB of disk quota 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, tick the *No Route* box.
- d. (Optional) If you would like to create a route for your app, leave the *No Route* box unticked and choose a host name (if different than your app name) and a domain for your app.

##### i 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 \[page 984\]](#)

[Create Routes \[page 984\]](#)

## 5.2.3.2 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</a> page	For the application that you'd like to start or stop, choose the respective icon from the <a href="#">Actions</a> column: <ul style="list-style-type: none"><li>○ <b>▶ (Start)</b> - This starts the first instance of your application and makes it available to users.</li><li>○ <b>◉ (Stop)</b> - This stops all running instances of the application.</li></ul> You cannot directly restart an application from this page.
<a href="#">Overview</a> page of your applica- tion	<p><b>i Note</b></p> <p>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>Choose the button corresponding to the desired action:</p> <ul style="list-style-type: none"><li>○ <b>Start</b> - This starts the first instance of your application and makes it available to users.</li><li>○ <b>Stop</b> - This stops all running instances of the application.</li></ul>

Page	Actions
	<ul style="list-style-type: none"><li>○ <a href="#">Restart</a> - This restarts the application.</li></ul> <div style="border-left: 3px solid #0070C0; padding-left: 10px;"><b>i Note</b><p>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.</p><p>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></div>

## Related Information

[Deploy an Application \[page 980\]](#)

### 5.2.3.3 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 \[page 980\]](#)

## 5.2.3.4 Configuring Application URLs

By default, all applications running on SAP Cloud Platform 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 Cloud Platform 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.

### i 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 \[page 987\]](#)

[About Routes in the Cockpit \[page 984\]](#)

### 5.2.3.4.1 About Routes in the Cockpit

Routes are the URLs that enable your end users to reach your application.

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.

Routes belong to a space, and therefore are managed at space level. Currently, you can create, map, and delete HTTP routes. An HTTP route includes a domain, a host name (or subdomain), and an optional path. 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.

The number of routes you can create in a space depends on your subaccount entitlements and quotas, or on the quota plan assigned to that space (if such a quota plan exists). The maximum number of routes is determined through the Application Runtime quota: 1 GB of Application Runtime comes with 10 routes.

#### Related Information

[Create Routes \[page 984\]](#)

[Map Routes to Applications Using the Cockpit \[page 986\]](#)

### 5.2.3.4.2 Create Routes

You can configure the URLs through which end users can reach your applications.

#### 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.

The following steps guide you through the procedure of creating routes by using the SAP Cloud Platform cockpit, but you can also create routes by using the CF CLI. For more information, see <https://cli.cloudfoundry.org/en-US/cf/create-route.html>.

## Procedure

1. Navigate to your Cloud Foundry space.
2. Choose *Routes* from the left hand-side navigation.
3. Choose *New Route* to create a route.
4. In the dialog, enter the following parameters:

Parameter	Details
<b>Domain</b>	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.  From the dropdown menu, you can choose either a shared domain or a private domain that you've previously created using the CF CLI.  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> .
<b>Host Name</b>	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>
<b>Path</b>	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: <code>https://&lt;host name&gt;.&lt;domain&gt;/&lt;path&gt;</code>

You can see the preview of your route at the bottom of the dialog.

5. When you're happy with your route, choose *Save*.

## Next Steps

Once you've created a route, you must map it to your application. Additionally, you also have the option to bind it to a route service instance, by choosing  (Bind Route Service) from the *Actions* column.

## Related Information

[Map Routes to Applications Using the Cockpit \[page 986\]](#)

## 5.2.3.4.3 Map Routes to Applications Using the Cockpit

Once a route has been created, you can 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.

### Procedure

1. Navigate to the [Routes](#) page in your Cloud Foundry space.
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 2 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

[Create Routes \[page 984\]](#)

## 5.2.3.4.4 Using Custom Domains

SAP Cloud Platform allows subaccount owners to make their SAP Cloud Platform 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:

## 5.2.4 Make Your Custom Identity Provider Visible in the Login Screen

Add your custom identity provider to the login screen of the Cloud Foundry environment of SAP Cloud Platform.

### Context

When you're 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 to a login screen. By default the login screen shows your account on the SAP ID Service. To make an account of your custom Identity provider (IdP) visible, enter a URL with the `discoveryPerformed` and `login_hint` parameters. Replace the variable values (`<region>` and `<origin>`) with your own.

#### i Note

Call the URL for every browser that you use.

### Procedure

1. Open your browser and call the URL:

```
https://login.cf.<region>.hana.ondemand.com/login?  
discoveryPerformed=true&login_hint=%257B%2522origin%2522%3A%2522<origin>%2522%257D
```

#### ❖ Example

```
https://login.cf.eu10.hana.ondemand.com/login?  
discoveryPerformed=true&login_hint=%257B%2522origin%2522%3A%2522acme_company%2522%257D
```

2. Sign in to your custom IdP.

3. (OPTIONAL) Test the result.
  - a. Choose your e-mail account in the upper right corner.
  - b. Choose *Sign Out*.
  - c. Call the initial URL of the application that you wanted to log in, to check if you can now choose between the default SAP ID Service account and your account from your custom IdP.

## Related Information

[Log On with a Custom Identity Provider to the Cloud Foundry Environment Using the Cloud Foundry Command Line Interface \[page 933\]](#)

## 5.2.5 Audit Logging in the Cloud Foundry Environment

In this section you can find information for audit log functionalities in the SAP Cloud Platform Cloud Foundry environment.

## Related Information

[Audit Log Retrieval API Usage for the Cloud Foundry Environment \[page 988\]](#)

[Audit Log Retention for the Cloud Foundry Environment \[page 991\]](#)

[Audit Log Viewer for the Cloud Foundry Environment \[page 992\]](#)

### 5.2.5.1 Audit Log Retrieval API Usage for the Cloud Foundry Environment

The audit log retrieval API allows you to retrieve the audit logs for your SAP Cloud Platform Cloud Foundry environment account. It provides the audit log results as a collection of JSON entities.

The audit log retrieval API is protected by OAuth, and to consume it a valid OAuth client needs to be generated and used by the client system.

Retrieving of audit logs via the Audit Log Retrieval API is limited to the size of the audit logs generated for the account.

## Prerequisites

The user executing the procedure is expected to have the `Space Developer` role for the corresponding Space.

### i Note

For security reasons it is strongly recommended the auditlog-management Service-Instances are recreated at least every 90 days.

## Create instance of the "auditlog-management" service

1. Create a Cloud Foundry Org and Space, in case you do not have any. See [Create Spaces \[page 922\]](#).
2. Login to 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"  
`cf create-service auditlog-management default <SERVICE_INSTANCE>`
4. Create a key for the service instance:  
`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 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 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.

### i Note

The token is valid for 12 hours.

### i Note

The content of the access token can be displayed using a standard JWT decoder

## Audit Log Retrieval

The Audit Log Retrieval API supports server-side paging. If a given query produces a result with significant size, the result will be chunked. Then the response will contain an HTTP header with a handle, with which to retrieve the next chunks of the result. The handle can be passed via URL parameter in the subsequent retrieval request.

Supported request parameters:

- `time_from` and `time_to` – if no time filter is specified the default timeframe of 30 days back is returned. The time should be provided in the following format: 2018-05-11T10:42:00. Times are UTC.
- `handle` – in case the result set is too big, it will be chunked, and a handle will be returned for reading the next chunk. It is returned as a Response Header in the form: `Paging: handle=<value>`. Then `handle=<value>` can be provided as a request parameter in the subsequent retrieval request.

### 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?  
handle=2018-06-14T10:11:18.968%3C4f932695-8616-4e1f-ac9a-1cdfb758f01d  
<url_from_service_key>/auditlog/v2/auditlogrecords?handle=<handle_value>
```

Response codes:

HTTP Codes

HTTP 200 OK

HTTP 204 NO\_CONTENT

HTTP 401 UNAUTHORIZED

HTTP 400 BAD\_REQUEST

HTTP 403 FORBIDDEN

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-e5b3-4cc5-8c90-964165axXXXX",  
  "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](#)

### 5.2.5.2 Audit Log Retention for the Cloud Foundry Environment

The audit log data stored for your account will be retained for 30 days, after which it will be deleted.

In case you want to retain and use the data for more than 30 days you can retrieve it via the [Audit Log Retrieval API Usage for the Cloud Foundry Environment \[page 988\]](#) and store it in another persistent storage..

## 5.2.5.3 Audit Log Viewer for the Cloud Foundry Environment

The SAP Cloud Platform Audit Log Viewer displays the audit logs for your Cloud Foundry account, produced by SAP applications and services you've subscribed to.

The UI allows you

- read the audit logs written for your account in selected time frame,
- display more detailed information on each single audit log record,
- filter the retrieved client-side chunk for certain strings,
- 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.

### Audit Log Viewer subscription

To use the Audit Log Viewer you have to subscribe for it using the SAP Cloud Platform cockpit in the *Subscriptions* tab of your subaccount. Once you have subscribed, select *Go to Application* to you open the Audit Log Viewer and login there.

### Audit Log Viewer access

To retrieve the audit logs for your subaccount using the Audit Log Viewer you need to have proper authorizations. See <https://docs.cloudfoundry.org/concepts/roles.html#permissions>. Create a RoleCollection, include the *auditlog-viewer!t\*/Auditlog Auditor* role and the *auditlog-management!b\*/Auditlog Auditor* role. Assign it to a user or create a rule to assign it to users based on the SAML Assertion coming from the IDP.

#### i Note

Only account members with the Security Administrator role are authorized to edit application authorizations.

## 5.3 Administration and Operations in the ABAP Environment

Learn about the different account administration and application operation tasks which you can perform in the ABAP environment.

If you use the ABAP environment, you must perform the account administration tasks in the Cloud Foundry environment. See:

- [Account Administration in the Cockpit \[page 754\]](#)
- [Org Administration Using the Cloud Foundry CLI \[page 930\]](#)

For application operations in the ABAP environment, see:

- [Application Operations in the ABAP Environment \[page 993\]](#)

### 5.3.1 Application Operations in the ABAP Environment

Learn more about how you can use apps to:

- 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

#### i Note

For an overview about the required business catalogs and business roles to access applications in the ABAP environment, see [Business Catalogs and Business Roles \[page 994\]](#). See also [How to Create a Business Role from a Template \[page 1045\]](#).

### Related Information

[Identity and Access Management \[page 1038\]](#)

[Business User Management \[page 1036\]](#)

[Maintain Certificate Trust List \[page 1064\]](#)

[Communication Management \[page 1017\]](#)

[Software Component Lifecycle Management \[page 1068\]](#)

### 5.3.1.1 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 ID	Business Catalog ID
Employee Master Data	<a href="#">Maintain Employees [page 1036]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_HCM_BC_EMP_MDT_PC <i>Employee - Master Data</i>
User Interface Configuration	Manage Launchpad Pages	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI_FLD <i>User Interface - Fiori Launchpad Design</i>
User Interface Configuration	Manage Launchpad Settings	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI <i>User Interface - Configuration</i>
User Interface Configuration	Manage Launchpad Spaces	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_UI_FLD <i>User Interface - Fiori Launchpad Design</i>
Message Monitoring	<a href="#">Assign Recipients to Users [page 1060]</a>	SAP_BR_CONF_EXPERT_BUS _NET_INT Configuration Expert - Business Network Integration	SAP_CA_BC_COM_CONF_PC <i>Communication Management - Message Monitoring Configuration</i>
Message Monitoring	Message Monitoring Dashboard	SAP_BR_CONF_EXPERT_BUS _NET_INT Configuration Expert - Business Network Integration	SAP_CA_BC_COM_ERR_PC <i>Communication Management - Message Monitoring and Error Handling</i>
Message Monitoring	Message Monitoring Overview	SAP_BR_CONF_EXPERT_BUS _NET_INT Configuration Expert - Business Network Integration	SAP_CA_BC_COM_ERR_PC <i>Communication Management - Message Monitoring and Error Handling</i>
ABAP Test Cockpit	<a href="#">ABAP Test Cockpit Configurator [page 1000]</a>	BR_ADMINISTRATOR_SW_DE V Administrator - Software Development	SAP_CORE_BC_ATC_CONFIG <i>ABAP Test Cockpit Configurator</i>

<b>Area</b>	<b>Application</b>	<b>Business Role ID</b>	<b>Business Catalog ID</b>
Business Configuration	<a href="#">Business Configuration Change Logs [page 1013]</a>	BR_BPC_EXPERT <i>Configuration Expert - Business Process Configuration</i>	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Business Configuration		BR_BPC_EXPERT <i>Configuration Expert - Business Process Configuration</i>	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Business Configuration	<a href="#">Manage Number Range Intervals [page 1014]</a>	BR_BPC_EXPERT <i>Configuration Expert - Business Process Configuration</i>	SAP_CA_BC_IC_LND_PC <i>Business Configuration - Customizing</i>
Application Jobs	<a href="#">Application Jobs [page 1002]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_APJ <i>Application Jobs</i>
Identity and Access Management	<a href="#">Business Catalogs [page 1052]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>
Identity and Access Management	<a href="#">Business Role Templates [page 1055]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RM <i>Role Management</i>
Identity and Access Management	<a href="#">Display Authorization Trace [page 1057]</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 Connectivity Trace [page 1025]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_RA <i>Role Assignment</i>
Identity and Access Management	<a href="#">Display Restriction Types [page 1051]</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 1052]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAM_UM <i>User Management</i>

<b>Area</b>	<b>Application</b>	<b>Business Role ID</b>	<b>Business Catalog ID</b>
Identity and Access Management	<a href="#">IAM Information System [page 1054]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	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 1043]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	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 1038]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</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 Deleted Business Users [page 1041]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_IAM_UMD <i>Identity and Access Management - User Management of Deleted Users</i>
Identity and Access Management	<a href="#">IAM Key Figures [page 1056]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_IAM_UM <i>User Management</i> SAP_CORE_BC_IAM_RA <i>Role Assignment</i> SAP_CORE_BC_IAM_RM <i>Role Management</i>
Security	<a href="#">Maintain Certificate Trust List [page 1064]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Maintain Protection White-lists [page 1065]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_SEC <i>Security</i>
Security	<a href="#">Manage Content Security Policy [page 1066]</a>	SAP_BR_ADMINISTRATOR <i>Administrator</i>	SAP_CORE_BC_SEC <i>Security</i>

<b>Area</b>	<b>Application</b>	<b>Business Role ID</b>	<b>Business Catalog ID</b>
Communication Management	<a href="#">Communication Arrangements [page 1021]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Display Communication Scenarios [page 1018]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Maintain Communication Systems [page 1023]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Maintain Communication Users [page 1019]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Communication Management	<a href="#">Maintain SAP Cloud Platform Extensions [page 1026]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_COM <i>Communication Management</i>
Output Management	<a href="#">Maintain Print Queues [page 1061]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_OM_PRT <i>Output Management - Printing</i>
Additional Software	<a href="#">Install Additional Software [page 1001]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_IAS <i>Additional Software</i>
Custom Code Migration	<a href="#">Custom Code Migration [page 1027]</a>	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 1071]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_A4C_BC_MSCL_PC <i>Lifecycle Management - Software Components</i>
Development	<a href="#">SQL EXPLAIN [page 1035]</a>	SAP_BR_DEVELOPER Developer	SAP_A4C_BC_MSCL_PC <i>Development - ABAP Development Tools</i>
Development	<a href="#">SQL Trace Analysis [page 1035]</a>	SAP_BR_DEVELOPER Developer	SAP_A4C_BC_MSCL_PC <i>Development - ABAP Development Tools</i>
Technical Monitoring	<a href="#">SQL Statement Analysis [page 1067]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_TMC <i>Technical Monitoring</i>
Technical Monitoring	<a href="#">Technical Monitoring Cockpit [page 1067]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_TMC <i>Technical Monitoring</i>

Area	Application	Business Role ID	Business Catalog ID
ABAP Dictionary	<a href="#">Manage Database Cache Configuration [page 999]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_A4C_DIC <i>Administration - ABAP Dictionary</i>
ABAP Dictionary	<a href="#">Repair CDS Views [page 998]</a>	SAP_BR_ADMINISTRATOR Administrator	SAP_CORE_BC_A4C_DIC <i>Administration - ABAP Dictionary</i>

### 5.3.1.2 Business Catalogs for Development 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 Development Tasks

Business Role Catalog	Authorizations	Restrictions
<i>ABAP Development Tools</i>	ADT Development	No transport request management (transport tasks only)
SAP_A4C_BC_DEV_PC	SQL EXPLAIN SQL Trace Analysis	
<i>Transport Management</i>	ADT Transport Management	No release of transport requests and no customizing requests
SAP_A4C_BC_TRN_MNG_PC		
<i>Transport Release Management</i>	ADT Transport Release Management	No customizing requests
SAP_A4C_BC_TRN_REL_PC		
<i>Custom Apps and Services</i>	Maintain Business Roles	Only services that have their original in the current system
SAP_CORE_BC_EXT_TST		

### 5.3.1.3 ABAP Dictionary

#### 5.3.1.3.1 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, Error Category, Person responsible and 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.

### 5.3.1.3.2 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.

### **5.3.1.4 ABAP Test Cockpit**

#### **5.3.1.4.1 ABAP Test Cockpit Configurator**

SAP Fiori app for editing ATC configurations.

#### **Purpose**

The ABAP Test Cockpit (ATC) Configurator enables you to maintain ATC configurations using SAP Fiori launchpad.

#### **Key Features**

1. Assign a check variant.
2. Change priorities.
3. Set a default configuration.

#### **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 \[page 1001\]](#).

## **Component for Customer Incidents**

BC-DWB-TOO-ATF

#### **Supported Device Types**

- Desktop

## 5.3.1.4.1.1 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 \[page 1045\]](#). Choose the template `SP_BR_ADMINISTRATOR_SW_DEV`.
  - To create a role from scratch, see [How to Create a Business Role from Scratch \[page 1044\]](#). Choose `ABAP Test Cockpit Configurator` (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 \[page 1039\]](#).)

When you log on to SAP Fiori launchpad with the assigned business user, the tile `ABAP Test Cockpit Configurator` will be available.

## 5.3.1.5 Additional Software

### 5.3.1.5.1 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.

### 5.3.1.6 Application Jobs

With this app you can 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. You can plan regular jobs which keeps you free to concentrate on other tasks.

## 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

### i Note

The parameters under *Scheduling Information* cannot 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 job logs and job results by clicking on the icon in the *Log* or the *Results* column

- 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.

### i Note

Please note, that currently it is not possible to differentiate between the read only access and read-and-write access. If the business catalog *Application Jobs* has been assigned to a business role, a user with this business role has the full access to the app.

### 5.3.1.6.1 Creating a Job Catalog Entry and a Job Template in ADT

Follow these steps to create a Job Catalog Entry and a Job Template in ADT.

#### Procedure

1. Implement the business logic. You need to create a class which implements certain interfaces so that it is usable in an application job. For more information, see [Implementing the Business Logic \[page 1004\]](#).
2. Define the Job Catalog Entry. With a method of a certain framework class, you create a Job Catalog Entry which refers to the class of step 1. For more information, see [Defining the Job Catalog Entry \[page 1005\]](#).
3. Define the Job Template. With another method of the framework class, you create a Job Template which refers to the Job Catalog Entry of step 2. For more information, see [Defining the Job Template \[page 1007\]](#).

#### Related Information

- [Setting up the Authorizations \[page 1006\]](#)  
[Scheduling an Application Job \[page 1006\]](#)

## 5.3.1.6.2 Implementing the Business Logic

The following development steps require a user with the development role.

Create a class which implements the two interfaces:

- **IF\_APJ\_DT\_EXEC\_OBJECT**
- **IF\_APJ\_RT\_EXEC\_OBJECT**

This class is considered the main class per application job. This class needs to be part of a customer-defined software component (not ZLOCAL) in order to be able to transport the class into subsequent systems.

The first interface, IF\_APJ\_DT\_EXEC\_OBJECT, contains design-time related methods. Method **GET\_PARAMETERS()** is called by the application jobs framework to get all supported selection parameters.

The method GET\_PARAMETERS() returns two internal tables:

- The content of the table **ET\_PARAMETER\_DEF** determines the parameter section for the job catalog entry, which will refer to this main class.
- The content of the table **ET\_PARAMETER\_VAL** determines the default values for these parameters in the job template, which will refer to the job catalog entry mentioned above.

The second interface, IF\_APJ\_RT\_EXEC\_OBJECT, contains runtime related methods. Method **EXECUTE()** is called by the application jobs framework if a scheduled job will actually be executed. It receives an internal table as parameter. This table is of the same type as the table ET\_PARAMETER\_VAL of method GET\_PARAMETERS(). The method EXECUTE() simply receives the parameters and values, which were stored in the template and which were originally delivered by the method GET\_PARAMETERS().

Please refer to the example code for an application jobs main class. Literals are used instead of language-dependent texts in order to simplify the example.

### ↳ Sample Code

```
CLASS zcl_test_apj_simple DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .
  PUBLIC SECTION.
    INTERFACES if_apj_dt_exec_object.
    INTERFACES if_apj_rt_exec_object.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_test_apj_simple IMPLEMENTATION.
  METHOD if_apj_dt_exec_object~get_parameters.
    " Return the supported selection parameters here
    et_parameter_def = VALUE #(
      ( selname = 'S_ID' kind = if_apj_dt_exec_object=>select_option
        datatype = 'C' length = 10 param_text = 'My
        ID' changeable_ind = abap_true )
      ( selname = 'P_DESCR' kind = if_apj_dt_exec_object=>parameter
        datatype = 'C' length = 80 param_text = 'My Description' lowercase_ind =
        abap_true changeable_ind = abap_true )
      ( selname = 'P_COUNT' kind = if_apj_dt_exec_object=>parameter
        datatype = 'I' length = 10 param_text = 'My
        Count' changeable_ind = abap_true )
      ( selname = 'P_SIMUL' kind = if_apj_dt_exec_object=>parameter
        datatype = 'C' length = 1 param_text = 'My Simulate Only' checkbox_ind =
        abap_true changeable_ind = abap_true )
    ).
    " Return the default parameters values here
  ENDMETHOD.

```

```

    et_parameter_val = VALUE #( 
        ( selname = 'S_ID' kind = if_apj_dt_exec_object->select_option sign
        = 'I' option = 'EQ' low = '4711' )
        ( selname = 'P_DESCR' kind = if_apj_dt_exec_object->parameter sign
        = 'I' option = 'EQ' low = 'My Default Description' )
        ( selname = 'P_COUNT' kind = if_apj_dt_exec_object->parameter sign
        = 'I' option = 'EQ' low = '200' )
        ( selname = 'P_SIMUL' kind = if_apj_dt_exec_object->parameter sign
        = 'I' option = 'EQ' low = abap_true )
    ).

ENDMETHOD.

METHOD if_apj_rt_exec_object~execute.
    TYPES ty_id TYPE c LENGTH 10.
    DATA s_id      TYPE RANGE OF ty_id.
    DATA p_descr   TYPE c LENGTH 80.
    DATA p_count   TYPE i.
    DATA p_simul   TYPE abap_boolean.
    " Getting the actual parameter values
    LOOP AT it_parameters INTO DATA(ls_parameter).
        CASE ls_parameter-selname.
            WHEN 'S_ID'.
                APPEND VALUE #( sign    = ls_parameter-sign
                                option  = ls_parameter-option
                                low     = ls_parameter-low
                                high    = ls_parameter-high ) TO s_id.
            WHEN 'P_DESCR'. p_descr = ls_parameter-low.
            WHEN 'P_COUNT'. p_count = ls_parameter-low.
            WHEN 'P_SIMUL'. p_simul = ls_parameter-low.
        ENDCASE.
    ENDLOOP.
    " Implement the job execution
ENDMETHOD.

ENDCLASS.

```

## Related Information

[Defining the Job Catalog Entry \[page 1005\]](#)

[Defining the Job Template \[page 1007\]](#)

[Setting up the Authorizations \[page 1006\]](#)

[Scheduling an Application Job \[page 1006\]](#)

### 5.3.1.6.3 Defining the Job Catalog Entry

Instead of a respective editor in ADT, Job Catalog Entries and Job Templates need to be created via a released API. This API shall be called from within a console application. The console application is only required in the development system.

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 package and transport request fit together (regarding transport layer) and that possible naming conventions are obeyed.

The Job Catalog Entry mainly contains the reference to the implementation class of the business logic.

### • Example

A code example of a console application that generates the minimal number of required development objects: One Job Catalog Entry and one related Job Template is shown in [Defining the Job Template](#) (see [Related Links](#) below).

## Related Information

[Defining the Job Template \[page 1007\]](#)

[Setting up the Authorizations \[page 1006\]](#)

[Scheduling an Application Job \[page 1006\]](#)

### 5.3.1.6.4 Scheduling an Application Job

Find out how to schedule an Application Job.

Open the Fiori app **Application Jobs** in the Fiori Launchpad and perform the following steps:

1. Click the + button.
2. Select the Job Template.
3. Maintain Scheduling Options.
4. Maintain Parameters.
5. Click Schedule.

You can find the log of the executed jobs on the entry screen of the Fiori app Application Jobs.

### 5.3.1.6.5 Setting up the Authorizations

Some further activities in ADT 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 in ADT \[page 1003\]](#), an object of the type **IAM App** is created automatically. It has the name **<job catalog entry name>\_SAJC**.

Create an IAM Business Catalog via ADT:

1. Right-click the package, where the already created objects are located.
2. New > Other ABAP Repository Object.
3. Collapse Cloud Identity and Access Management.
4. Double-click Business Catalog.
5. Enter a name (e.g. ZTEST\_MY\_SIMPLE\_JOBS) and a description.
6. Save and choose the appropriate transport request.

7. Select the tab Apps on the bottom.
  8. Press the button Add... on the top.
  9. Enter the name of the IAM App from above (ZTEST\_MY\_SIMPLE\_JOB\_SAJC).
  10. Save.
  11. Press the button Publish Locally on the top right.
2. 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 (e.g. ZTEST\_MY\_SIMPLE\_JOBS ) and a description.
  3. Click on Assigned Business Catalogs.
  4. Click Add.
  5. Select the business role created in the previous step (e.g. ZTEST\_MY\_SIMPLE\_JOBS).

**i Note**

Currently, a business user who shall be able to schedule application jobs also needs a business role, to which the business catalog Application Jobs (SAP\_CORE\_BC\_APJ) has been assigned. The Administrator role SAP\_BR\_ADMINISTRATOR is such a role, but it is recommended to add the business catalog Application Jobs (SAP\_CORE\_BC\_APJ) to the newly created role as well.

6. Save.
3. Assign the Business Role to a User.  
For more information, see [How to Maintain Business Users \[page 1039\]](#).

## Related Information

[Scheduling an Application Job \[page 1006\]](#)

### 5.3.1.6.6 Defining the Job Template

The creation of a Job Template follows the same technical rules as a Job Catalog Entry as described in [Defining the Job Catalog Entry \[page 1005\]](#).

The Job Template represents a set of default parameters for the assigned Job Catalog Entry. The Job Template is mandatory for the Fiori app Application Jobs to choose a job definition to be executed. A Job Catalog Entry can have more than one Job Template.

The following code example shows a console application that generates the minimal number of required development objects: 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(l_o_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.
  l_o_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_temp_val.
  NEW zcl_test_apj_simple( )->if_apj_dt_exec_object~get_parameters(
    IMPORTING
      et_parameter_val = lt_parameters
  ).
TRY.
  l_o_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.
```

## Related Information

[Setting up the Authorizations \[page 1006\]](#)

### 5.3.1.6.7 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.  
  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 name of existing job template !!!!  
      lv_template_name = 'ZTEST_MY_SIMPLE_JOB_TEMP'.  
    * immediate start  
    *ls_start_info-start_immediately = 'X'.  
    * alternatively with timestamp:  
      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 = 'Blabla 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.
* 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
        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 cancelled
* 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.
```

### 5.3.1.6.8 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.

#### i Note

Please note, that the jobs with the status *Finished* cannot be canceled because they have already been run.

## Related Information

[Application Jobs \[page 1002\]](#)

### 5.3.1.6.9 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
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.

#### i Note

This is the default behaviour. If the application chooses to have different schemes, this is documented in the application-specific documentation.

## 5.3.1.6.10 Glossary for Application Jobs

Overview of terminology used for application jobs

Terminology Overview

Term	Description	Area
Job chain	A series of jobs that should be performed in a specific sequence.	Application Jobs

## 5.3.1.7 Business Configuration

### 5.3.1.7.1 Business Configuration Change Logs

With this app you can keep track of content changes in your business configuration tables.

#### Key Features

You can use this app to:

- View which changes were made to the records of your business configuration tables.
- View when changes were made to the records of your business configuration tables.
- View who made changes to the records of your business configurations tables.

#### Supported Device Types

- Desktop
- Tablet

#### Component for Customer Incidents

If you need support or experience issues, please report an incident under component BC-CUS-TOL-ALO.

### 5.3.1.7.1.1 Viewing a Change Log

#### Prerequisites

To access the app, you need to have the following business catalog assigned to your user: SAP\_CA\_BC\_IC\_LND\_PC. This business catalog is contained in the business role template: SAP\_BR\_BPC\_EXPERT.

Additionally, the *Log Changes* flag 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.

#### i Note

If you experience any issues viewing the log, please refer to the following SAP Notes: [2523607](#) (How to check if logging is on) and [2724472](#) (Automatic checks and solution for most frequent issues in table logging).

#### Context

Find out how to view 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.

### 5.3.1.7.2 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.

For more information, see:

- [Maintaining Number Range Intervals \[page 1016\]](#)
- [Changing the Number Range Status \[page 1017\]](#)
- [Checking Properties of a Number Range Object \[page 1015\]](#)

### 5.3.1.7.2.1 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.

#### i 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. For more information, see [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](#)

### 5.3.1.7.2.2 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.

#### i 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.

### 5.3.1.7.2.3 Changing the Number Range Status

The *Number Range Status* or *Number Range Level* is the last number used by an interval.

#### i 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*.

### 5.3.1.8 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 user for inbound communication according to your needs using the *Maintain Communication Users* app.
2. Create 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.

## Related Information

[Maintain Communication Users \[page 1019\]](#)

[Maintain Communication Systems \[page 1023\]](#)

[IAM Information System \[page 1054\]](#)

### 5.3.1.8.1 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

## **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.

### **5.3.1.8.2 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

## **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 Systems \[page 1023\]](#)

[How to Create Communication Users \[page 1020\]](#)

### 5.3.1.8.2.1 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. Define the required password.

Either provide a password (basic authentication) or upload a certificate for certificate-based authentication.

#### i Note

We recommend that you use the *Propose Password* button.

4. If necessary, upload the required X.509 client certificate.
5. 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 \[page 1019\]](#)

[Maintain Communication Systems \[page 1023\]](#)

### **5.3.1.8.3 Communication Arrangements**

You can use this app to create and edit communication arrangements. Communication arrangements help you to configure the electronic data exchange between the system and a communication partner.

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.

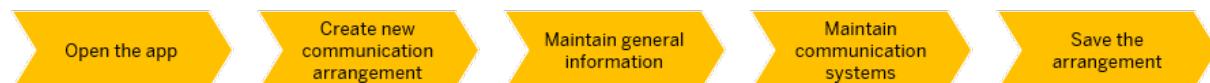
## 5.3.1.8.3.1 How to Create a Communication Arrangement

### Prerequisites

- You have already created a communication system with inbound and outbound users. For more information, see section *Maintain Communication Systems*.
- You have already created a communication user with a supported authentication type that is defined in the selected communication scenario. For more information, see section *Maintain Communication Users*.

### Context

#### Process Steps



To create a communication arrangement, proceed as follows:

### Procedure

1. Open the *Maintain 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. New screen *Communication Arrangements* is now open. Under *Common Data* in the field *Communication System* select a communication system that you want your system to connect to.
4. If the selected communication system already has a communication user for inbound communication, the user and required authentication type will be displayed in the field *User Name* under *Inbound Communication*. In the *Inbound Services* section, the URLs to the service endpoints are displayed.
5. If the selected communication system already has a communication user for outbound communication, the user and required authentication type will be displayed in the field *User Name* under *Outbound Communication*.
6. Save the arrangement.

## 5.3.1.8.4 Maintain 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 \[page 1019\]](#)

## 5.3.1.8.4.1 How to Create Communication Systems

### Context

To create a communication system perform the following steps:

### Procedure

1. On the initial screen 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](#)
- [Technical Data](#)

Here you can define alternative IDs for the communication system: the logical system ID that is required for IDoc communication and the business system ID that identifies the communication system in your system landscape.

#### i Note

Please note, that you can't use a protocol as a host name.

- [Destination Service](#)

Because SAP Cloud Platform ABAP Environment uses destination services of SAP Cloud Platform 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](#).

#### i Note

A predefined instance is selected by default. If you want to add your own instance, deselect this instance.

- [Contact Information](#)

4. Add technical users for inbound and/or outbound 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 app [Maintain Communication User](#).

#### i Note

Inbound users are communication users provided by your system and are used by the communication system to call the inbound services. Outbound users are communication users provided by the external communication system to call the outbound services.

5. Save the changes.

You can now establish a communication arrangement with the created system. Use [Maintain Communication Arrangements](#) app for this purpose.

## Related Information

[Maintain Communication Systems \[page 1023\]](#)

### 5.3.1.8.5 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.

## **5.3.1.8.6    Maintain SAP Cloud Platform Extensions**

You can use this app to create SAP Cloud Platform extensions to build extension applications.

With this app you can create SAP Cloud Platform Extensions for your SAP S/4HANA system. These automatically connect the two systems and enable you to build extension applications for SAP S/4HANA on SAP Cloud Platform.

### **Key Features**

You can use this app to:

- Create new SAP Cloud Platform extensions
- Temporarily disable SAP Cloud Platform 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 Information**

[Triggering an Automated Integration Between SAP S/4HANA Cloud and SAP Cloud Platform](#)

## 5.3.1.8.7 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.

## 5.3.1.9 Custom Code Migration

### Purpose

The Custom Code Migration app enables you to analyze custom code that needs to be migrated from an SAP Business Suite system to SAP S/4HANA (on-premise). To evaluate the custom objects to be adopted, it performs the SAP S/4HANA custom code checks.

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

Scoping:

- You as an user define which ABAP custom code needs to be migrated to be taken over to SAP S/4HANA (based on usage data)
- This app creates a deletion transport in order to delete unused ABAP source code during the system conversion to SAP S/4HANA

Analysis of SAP S/4HANA custom code check findings:

- Analytical representation of SAP S/4HANA custom code check findings
- Results can be filtered by scope and usage data

## 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 \[page 1028\]](#)

## Component for Customer Incidents

BC-DWB-CEX

## Supported Device Types

- Desktop

### 5.3.1.9.1 Enable Usage of the Custom Code Migration App

You can use this app to perform SAP S/4HANA custom code checks in the on-premise system in which the custom code to be analyzed is stored.

## Prerequisites

- Your company has a global account for *SAP Cloud Platform*.
- You need permission to access the *SAP Cloud Platform Cockpit* and the *Cloud Connector* as well as to create subaccounts.

## Overview

Learn how to enable and implement the Custom Code Migration (CCM) app.

As an **SAP administrator**, you have to perform the following steps:

1. A global account member must create [subaccounts \[page 1029\]](#) for the following environments in the *SAP Cloud Platform Cockpit*:

1. [Neo environment](#)
2. [Cloud Foundry environment](#)
2. [Configure and administrate the Cloud Connector \[page 1029\]](#).
3. Set up [the destinations to enable connectivity \[page 1032\]](#) from ABAP environment to your on-premise systems.
4. In the [SAP Fiori launchpad](#) of your ABAP environment, you have to [administrate the user assignments to your project manager\(s\) and maintain the relevant communication arrangements \[page 1033\]](#).
5. In the on-premise system, you have to [install the remote stubs \[page 1034\]](#) for implementing the relevant function modules to access your custom code.

As a business user, this enables you to use the destination pointing to the on-premise system when creating a project in the CCM app.

## Creating Subaccounts

Your company will get a global account when a SAP Cloud Platform contract with SAP is signed.

You have to create the following subaccounts to run the `Custom Code Migration` app in your global account:

### Neo Environment

In the [SAP Cloud Platform Cockpit](#), [create a subaccount](#) in the Neo environment. This enables you to connect your cloud connector which contains the RFC connection to your on-premise system.

### Cloud Foundry

In the [SAP Cloud Platform Cockpit](#), [create a subaccount](#) in the Cloud Foundry environment. This enables you to create a space and a service instance.

## Cloud Connector: Configuration and Administration

### i Note

Ensure that the [cloud connector](#) is installed and configured.

### Connect the Neo Environment Subaccount to the Cloud Connector

For the [Cloud to On-Premise](#) connection, connect your Neo environment [subaccount](#) to your cloud connector.

### Configure the Access Control (RFC) for the Remote Connection

Define the access authorization by using the resource accessible, also known as access control list (ACL), for the connection to the checked system as follows:

1. In the [Cloud Connector Administration](#) of the SAP Cloud Platform Cockpit, configure the [access control list \(RFC\)](#).
2. [Define the permitted function modules](#) as resources for a particular back-end system.

## i Note

You can also import the Custom Code Migration scenario as a zip file in the *Cloud Connector*. For more information, see SAP Note [2861842](#) - Custom Code Migration in SAP Cloud Platform ABAP Environment: Set up SAP Cloud Connector.

To define the function modules manually, see the following list:

List of the Required Function Modules

### Function Modules

---

BAPI\_USER\_GET\_DETAIL

---

FUNCTION\_EXISTS

---

REPOSITORY\_ENVIRONMENT\_ALL

---

RFC\_GET\_NAMETAB

---

RFC\_GET\_SYSTEM\_INFO

---

RFC\_PING

---

RFC\_SYSTEM\_INFO

---

RFCPING

---

RS\_ABAP\_CHECK\_PROGRAM\_E

---

RS\_ABAP\_EXPAND\_DDL\_OBJ\_LIST\_E

---

RS\_ABAP\_EXPORT\_COMP\_PROCS\_E

---

RS\_ABAP\_GET\_ALL\_DYNPROS\_E

---

RS\_ABAP\_GET\_CODE\_RANGES\_E

---

RS\_ABAP\_GET\_DDIF\_TBL\_E

---

RS\_ABAP\_GET\_DDIF\_VIEW\_E

---

RS\_ABAP\_GET\_DDL\_DEPENDENCIES\_E

---

RS\_ABAP\_GET\_INSP\_PROGRAMS\_E

---

RS\_ABAP\_GET\_OBJECTS\_E

---

RS\_ABAP\_GET\_OBJECT\_CLASS\_E

---

RS\_ABAP\_GET\_OBJECT\_E

---

RS\_ABAP\_GET\_OBJECTS\_E

---

RS\_ABAP\_GET\_TYPE\_INFO\_E

---

## Function Modules

---

RS\_ABAP\_GET\_TYPES\_E

---

RS\_TOOL\_ACCESS\_REMOTE

---

SCA\_REMOTE\_DATA\_ACCESS

---

SCA\_REMOTE\_DATA\_VERSION

---

SCI\_ANALYZE\_SQL\_STMNTS

---

SCI\_DCTHDB\_DOWNLOAD

---

SCI\_F4\_EXIT\_CHK

---

SCI\_F4\_EXIT\_CHKV

---

SCI\_F4\_EXIT\_ERRTY

---

SCI\_F4\_EXIT\_INSP

---

SCI\_F4\_EXIT\_OBJS

---

SCI\_GET\_INSPECTION\_CODE\_STAT

---

SCI\_GET\_INSPECTION\_PLAIN\_LIST

---

SCI\_GET\_INSPECTION\_RESULT

---

SCI\_GET\_INSPECTION\_RESULTS

---

SCI\_GET\_OBJECTS\_PACKAGES

---

SCI\_INSPECT\_LIST

---

SCI\_PERF\_CHKLST\_COMPLIANCE

---

SCI\_QUERY\_SET\_GUI\_STATUS

---

SCI\_REMOTE\_DELETE

---

SCI\_REMOTE\_OBJLIST\_CHECK

---

SCI\_REMOTE\_OBJLIST\_RESULT\_SHOW

---

SCI\_REMOTE\_RESULT\_DISPLAY

---

SCI\_REMOTE\_SQL\_STMNTS\_ANALYZE

---

SCI\_REMOTE\_SQLT\_RESULT\_SHOW

---

SCI\_RUN

---

## Function Modules

SCI\_SHOW\_RESULTS  
SLINRMT\_RUN  
SUSG\_API\_ADMIN\_ITERATOR  
SUSG\_API\_DATA\_ITERATOR  
SUSG\_API\_ODATA\_ITERATOR  
SUSG\_API\_PROC\_ITERATOR  
SUSG\_API\_PROG\_ITERATOR  
SUSG\_API\_RDATA\_ITERATOR  
SUSG\_API\_SUB\_ITERATOR  
SVRS\_GET\_VERSION\_DIRECTORY\_46  
SYCM\_CALCULATE\_URLS  
SYCM\_CREATE\_DELETION\_TRREQ  
SYCM\_GET\_TADIR\_INFO  
SYCM\_HEALTH\_CHECK

## SAP Cloud Platform Cockpit: Setting Up the Destinations to Enable Connectivity to On-Premise Systems

### Creating a Space

Create a space in the Cloud Foundry subaccount for your organization or development activities.

For more information, see [Creating Spaces](#)

### Creating a Destination Service Instance

To define the applications in the ABAP environment that are needed for outbound connectivity, create a destination service instance and a service key in the created space for the destination service.

For more information, see

- [Creating a Destination Service Instance \(Optional\)](#)
- [Creating a Service Key for the Destination Service Instance \(Optional\)](#)

## Add an RFC Destination to Your Service Instance

To add a connection to the on-premise system for the CCM app, set up on-premise RFC connectivity for the ABAP environment by configuring a destination of type RFC.

For more information, see [Set Up an RFC Destination](#).

## SAP Fiori Launchpad: Administrate User Assignments and Maintain Communication Arrangements

### Administrat User Assignments

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. To do this, proceed as follows:

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.

#### i 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 Arrangements

To enable communication from your ABAP environment to your on-premise systems using Remote Function Calls (RFC), you need to create the following communication arrangements in your ABAP environment:

- [SAP Cloud Connector Integration](#) (SAP\_COM\_0200)
- [SAP CP CF Destination Service Integration](#) (SAP\_COM\_0276)

#### i Note

Use the [destination service instance \[page 1032\]](#) that you have created in the [SAP Cloud Platform Cockpit](#).

- SAP Custom Code Migration Integration (SAP\_COM\_0464)

#### i Note

To create these communication arrangements, you need the Administrator and the Project Manager – IT role.

SAP\_COM\_0464 can only be created when communication arrangements SAP\_COM\_0200 and SAP\_COM\_0276 have already been created.

To set up the RFC connection from the ABAP system in the ABAP environment to your on-premise system, proceed as follows:

1. In the *SAP Cloud Platform Cockpit*, create a destination to your on-premise system in the service instance.
2. Log on to the *SAP Fiori launchpad* of your ABAP environment.
3. In the *Communication Management* section, select the *Communication Arrangement* tile.
4. Create a new communication arrangement using the SAP\_COM\_0464 scenario.
5. If not yet available or defined, create a communication system for the communication arrangement to define an endpoint for your checked system.
  1. Use the *System ID* and *System Name* of the checked system.
  2. Choose *Create*.
  3. In the *General* tab, switch on the slider for the *Destination Service*.
  4. Select the service instance that is defined in communication arrangement SAP\_COM\_0276 as *Instance*.

Enter the corresponding [destination \[page 1033\]](#) to your on-premise system that you have defined in the service instance in your *SAP Cloud Platform Cockpit* as *Name*.

#### i Note

You have to enter the exact name of the destination for *Name*.

5. Confirm with *Save*.
6. Select the communication system created in step 5 for your communication arrangement and confirm with *Save*.

Now, you can use the communication arrangement as *Destination* 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](#) Custom Code Migration Fiori App: Remote Stubs for the Checked System.

## Related Information

[SAP Note 2436688](#)

[Custom Code Migration Guide for SAP S/4HANA 1909](#)

[How to check your custom ABAP code for SAP Cloud Platform ABAP Environment](#)

## **5.3.1.10 Development**

### **5.3.1.10.1 SQL EXPLAIN**

With this app you can display access plans for SQL statements.

#### **Key Features**

This app provides the following key features:

- Prepared access plan of an SQL statement that you entered
- Executed access plan of the SQL statement

#### **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.

### **5.3.1.10.2 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
- Navigation to the ABAP source code location in ADT
- Editing of the executed SQL statement

- View and recalculation of the access plan of the SQL statement

## 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.

### 5.3.1.11 Employee Master Data

#### 5.3.1.11.1 Business User Management

Business User Management enables and supports the lifecycle maintenance of business users.

#### Blocking of Business Partner Data in Business User Management

For information about the blocking of business partner data, see the Business Partner documentation on SAP Help Portal at  [SAP S/4HANA Cloud](#)  [Master Data](#)  [Business Partner/Customer/Supplier](#)  [Scheduling Block Business Partners](#) .

#### 5.3.1.11.2 Maintain Employees

Create employees and modify employee information.

#### Prerequisites

The catalog SAP\_HCM\_BC\_EMP\_MDT\_PC is assigned to the administrator role. The assignment enables the [Maintain Employees](#) application in the launchpad. Use the [Maintain Business Users](#) application to assign the role to a user ID.

With this app you can create employees and modify employee information.

## Key Features

You can use this app to:

- Create employees
- Modify employee information such as personal data (Last Name, First Name, and E-Mail), and employee data (Employee ID, Valid From, Valid To)
- Display the changes that have been made to an employee
- Search for employee details providing the employee ID
- Mass upload employees

## 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.

### 5.3.1.11.3 How to Delete an Employee

#### Context

If you delete an employee in [Maintain Employees](#), it is not physically deleted from the database but is marked for archiving. This is contrary to the Fiori app [Maintain Business Users](#), where the user is physically deleted.

An employee cannot be deleted immediately for business process reasons. It is also not possible to delete an employee in the Fiori app [Maintain Employees](#), if the employee has a user assigned to it. In this case, you have to delete the user in the Fiori app [Maintain Business Users](#) first.

By default, all employees that are marked for archiving (deleted) are hidden. You can unhide them and also undo the mark (deletion) to reuse the employee. To do so, follow these steps:

## Procedure

1. In the Fiori app *Maintain Employees*, choose *Filters*.
2. Choose *More Filters*, select *Deleted* and choose *OK*.
3. In the dropdown menu for *Deleted*, select *YES* or leave it blank, and choose *OK*.
4. Search for a deleted employee.

You may have to change the settings of the table to add the optional column *Deleted*.

5. Choose a deleted employee to get to the *Display Employee* view.
6. Choose *Undo Deletion*.

The archiving flag will be removed from the employee.

You can now use the employee again, for example, to create a new user for it.

### 5.3.1.12 Identity and Access Management

The Identity and Access Management apps secure the access to your solution for your business users.

#### Related Information

[Business Catalogs \[page 1052\]](#)

[IAM Information System \[page 1054\]](#)

[Business Role Templates \[page 1055\]](#)

### 5.3.1.12.1 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

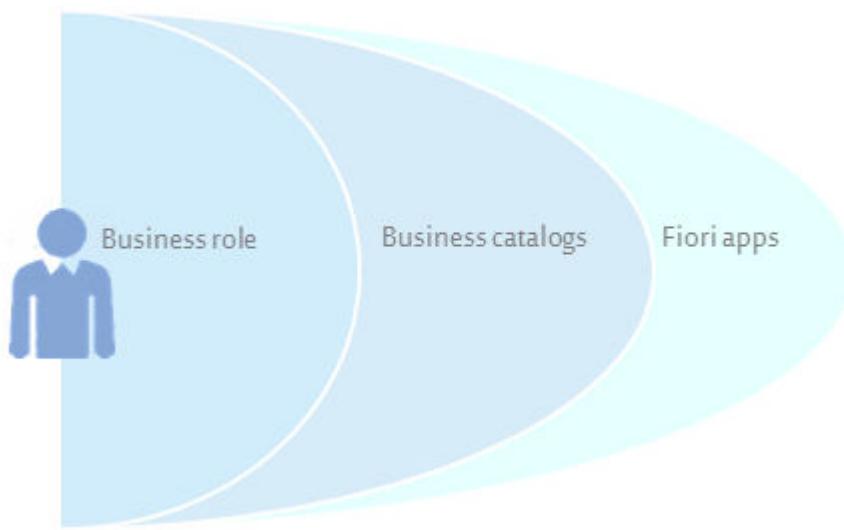
## **Related Information**

[Business Catalogs for Identity and Access Management Apps \[page 1058\]](#)

### **5.3.1.12.1.1 How to Maintain Business Users**

#### **Context**

You can change user data (for example, the user name) and regional settings (for example, the date and time format).



How users get access to SAP Fiori apps

## 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.

- **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.

### i Note

The CSV file needs to be UTF-8-encoded and must comply with the following pattern:

User Name	Role ID
<User name 1>	<Business role ID 2>
<User name 1>	<Business role ID 6>
<User name 2>	<Business role ID 2>
<User name 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.

- **Downloading User Lists**

- To download a list of all users with an email address as csv files for upload to SAP Cloud Platform 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 of the SAP Cloud Platform Identity Authentication.
  - c. To upload the csv file, go to ► *User Management* ► *Import Users* ▶.

### i Note

Make sure that the csv file is not opened with any program before the upload.

- Changing the Language Setting for a Business User
- To change the language setting for a business user, select it for editing and choose the required language in the *Regional Settings* part of the screen.

## 5.3.1.12.2 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.

## 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.

### 5.3.1.12.2.1 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.

#### i 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](#).

### i 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.

### 5.3.1.12.3 Maintain Business Roles

You can use this app to create and edit business roles, add business catalogs to the roles, and maintain access restrictions.

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. 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:

- 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

## 5.3.1.12.3.1 How to Create a Business Role from Scratch

Get an overview of how to create a 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.

### 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*.

#### i 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 [How to Create a Space and Page for a Business Role](#).
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.

### i 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.

## 5.3.1.12.3.2 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. 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.

### i Note

SAP does not recommend using them in their full scope, as the business catalogs might even conflict from a business perspective. Instead, adjust them to the tasks of the role in your company by choosing only the relevant business catalogs.

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, select *Create and Assign Launchpad Space Based on Space Template* and define the new space ID (for more information, see [How to Create a Space and Page for a Business Role](#)). Click **OK**.
3. A template already contains one or more business catalogs that 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.

**i 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

[Business Role Templates \[page 1055\]](#)

### 5.3.1.12.3.3 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. 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.

### **i** 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

## 5.3.1.12.3.4 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:

### Procedure

1. Open the business role you need to edit.
2. 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.

3. Make the required changes to the business role.
4. 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.

## 5.3.1.12.3.5 How to Maintain Restrictions

### Context

By maintaining restrictions you can define the subset of all existing business objects a user can view or edit when working with this business role. The access restrictions allow you to differentiate your business roles on a fine-granular level. When you specify for a business role that a certain object should not be visible or editable, this applies to all apps included in this role.

If you assign multiple business roles that contain the same business catalogs, but different access restrictions, all restrictions are aggregated to the business user.

Using the [Maintain Business Roles](#) app, you have the following options for maintaining restrictions:

## Procedure

- On the [Maintain Restrictions](#) tab, you can maintain restrictions for business roles.

The business catalog defines which access categories are available for maintaining and for which fields restriction values can be maintained. The following access categories can be available:

- [Value Help](#) (value help access)
- [Read, Value Help](#) (read access)
- [Write, Read, Value Help](#) (write access)

The business role aggregates these restrictions.

The available restriction fields represent the authorization-relevant attributes of the business objects that are used in a role and for which instanced based access restrictions for value help, read and write access can be granted (for example for a particular sales area).

Please note that not all restriction fields support number ranges. You can display the information whether a restriction field supports number ranges in the [Display Restriction Types](#) app. For more information, see [Display Restriction Types](#) (Related Information).

Depending on the required authorization concept for the business role, you can restrict data access for the following separately:

- [Value Help](#)
- [Read, Value Help](#)
- [Write, Read, Value Help](#)

### i Note

When you set an access category to [Restricted](#) and it is not included implicitly according to the rule above, you either need to maintain a particular field value or you need to grant unrestricted access ("\*[No Access](#)") for this restriction field.

- Value Help**
- You can define access restrictions for value helps that are used in a business role. Leaving fields empty will typically mean [No Access](#) for this restriction field.

These value help restrictions will not influence the defined restrictions for read access.

In the context of a business role, you can restrict the value help access, for example to business partners that belong to certain authorization groups.

- Read / Write**
- As a default, read access is unrestricted, and write access is not granted to business roles. You can add or remove restrictions or you can choose [Unrestricted \(\\*\)](#) in cases where you want to grant full access for a field.

Switching the read or write access to [Restricted](#) allows you to define which data can be seen and edited by the users assigned to this business role.

- In the [Restrictions and Values](#) section, you can define the instance-based restrictions for the desired restriction fields used for value helps.

## Example

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.

- **Leading Restriction**

For general organizational fields that are used in several different restriction types, you can define a restriction as leading. That means the value in this field is automatically passed on to other restriction types the field is used in as well.

Select the required field under *Restriction and Values* *General* and click the pencil icon. In the dialog box that opens, select the values you want to define as leading and switch *Leading Restriction* on. These values are then automatically transferred to the other restriction types the field used in.

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* (Country Template AU) and *CH01* (Country Template CH) and switch *Leading Restriction* on. Then these values are automatically distributed to all occurrences of the *Company Code* field.

## Related Information

[Display Restriction Types \[page 1051\]](#)

### 5.3.1.12.3.6 How to Download and Upload Business Roles

Download business roles from the test system and then upload them to the productive system to make them available there.

## Context

You can download one or more business roles from the test system and then upload them to the productive system using an `XML` file.

## Procedure

1. To download the required business roles, go to your test 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 productive system and click *Upload*.
4. Browse for the `XML` file and click *OK*.

## 5.3.1.12.3.7 How to Create a Space and Page for a Business Role

### Context

When creating a new business role, you can create a new launchpad space or use an existing launchpad space. For more information, see [How to Create a Space and Page for a Business Role](#).

## 5.3.1.12.4 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.

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.

## 5.3.1.12.5 Display Technical Users

This app shows all technical users that exist in the system.

With this app 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.

### Key Features

You can use this app to:

- Lock or unlock the following types of technical users: Print users, communication users, and initial user (`SAP_CUST_INI`) that comes with the new system
- Change user name and password of some types of technical users

### Supported Device Types

- Desktop
- Tablet

## 5.3.1.12.6 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 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.

## 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.

### 5.3.1.12.6.1 How to Handle Deprecated Business Catalogs

#### 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.

#### i 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 (obsolete) 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 two more releases before being deleted. During these two releases 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 1811 release. The business catalog will then be deleted with the Cloud 1905 release.

### 5.3.1.12.7 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. For example, you can use this app to check if a business user is using a particular app and to check which authorizations he or she has.

If you want to look up more information about a 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 as part of your daily work as SAP S/4 HANA Cloud administrator.

#### Key Features

You can use this app to:

- Check the usage of the following entities and how they are related: Business role, usiness role template, application, business user, business catalog, restriction
- 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.

#### 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.

## 5.3.1.12.8 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

### 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 Business Role from Scratch \[page 1044\]](#)

[How to Create a Business Role from a Template \[page 1045\]](#)

## 5.3.1.12.9 IAM Key Figures

With this app you can get a quick overview of critical issues of your business users and business roles. You can, for example, check if business users are locked or have too many business roles assigned. The relevant information is displayed in detailed charts so you can view the required figures at a glance.

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

### 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.

## 5.3.1.12.9.1 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.

## 5.3.1.12.10 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

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.
Filtered	When reading an object, an authorization check is taking place and certain data is filtered out defined by a DCL (Data Control Language).

If an authorization check resulted in a *Filtered* status, you can check which business roles expose the affected restriction type. One potential solution is 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.

### 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.

## 5.3.1.12.11 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. Currently, four business catalogs are supported.

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

## 5.3.1.12.12 Glossary for Identity and Access Management

Overview of 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.

## 5.3.1.13 Message Monitoring

### 5.3.1.13.1 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 is a technical user who can see messages that are in process or that had a technical error.
- 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.

## 5.3.1.14 Output Management

### 5.3.1.14.1 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
- Create a user for SAP Cloud Print Manager
- Reassign a print item to another print queue

#### i Note

The **Maintain Print Queues** app is intended as a tool for administrators to manage print queues and to correct any errors in the printing process. End users should not have access to it, as no business-related authorizations are checked in the app. A download or preview of the print queue items is thus not offered in the app.

#### 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

### 5.3.1.14.1.1 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 the customer's 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, you have to install *SAP Cloud Print Manager* in the customer's network from the [Install Additional Software](#) app so that you can regularly check via a connection to the backend system whether new print items are in the print queue. If this is the case, SAP Cloud Print Manager retrieves these items and sends them to the locally configured printer.

### 5.3.1.14.1.2 Working in the Maintain Print Queues App

#### 5.3.1.14.1.2.1 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. Enter the required information:
  - **Queue:** Print queue name
  - **Description:** Print queue description
  - **Format:** Print queue format
  - **Print User:** Technical user with which SAP Cloud Print Manager logs on to the system. You can connect several queues to the same *SAP Cloud Print Manager* - in this case all queues must use the same print user.
  - **Retention:** Spool Retention Period
4. Click [Create](#) to create a new print queue.

## Results

You have created a print queue you can use in the SAP Cloud Print Manager.

### i 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.

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

### 5.3.1.14.1.2.2 Creating Print Users

You want to create a print user for a print queue.

## Context

A print user is a technical user used by the SAP Cloud Print Manager to log on to the system and retrieve the print queue information.

## Procedure

1. Create a print queue as described in [Creating Print Queues \[page 1062\]](#).
2. Fill in the required information and select **New Print User** in the *Create Print Queue* dialog.
3. Enter the required information.
  - **Name:** Print queue name
  - **Description:** Print queue description
  - **Password:** Enter a password or select *Propose Password* to use a generated password.

### i Note

This password is required later on to configure to the *SAP Cloud Print Manager*.

4. Click *Create* to create a new print user.

## Results

You have created a new print user.

## Related Information

### 5.3.1.15 Security

#### 5.3.1.15.1 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 among 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
- 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.

### 5.3.1.15.2 Maintain Protection Whitelists

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 the clickjacking protection whitelist. 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.

In a second whitelist for trusted network zones, you can list URL patterns for which a redirect is allowed or blocked.

#### Key Features

You can use this app to:

- Add trusted hosts to the [Clickjacking Protection](#) whitelist (name, schema, port) or URL patterns to the [Trusted Network Zones](#) whitelist
- 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.

### **5.3.1.15.3 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 a whitelist of allowed 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 a whitelist of allowed sources
- Add new allowed sources to the whitelist  
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.

#### **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.

## **5.3.1.16 Technical Monitoring**

### **5.3.1.16.1 Technical Monitoring Cockpit**

With this app you can monitor SAP systems in the ABAP environment.

#### **Key Features**

This app provides the following key features:

- Monitoring of important technical information such as CPU utilization, configuration, memory, network, performance, storage, time spent, and workload for the application server ABAP and the SAP HANA database
- SQL statement analysis and SQL EXPLAIN for the SAP HANA database
- Analysis of SQL traces triggered on the application server ABAP

#### **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.

## **5.3.1.16.2 SQL Statement Analysis**

With this app you can analyse the performance of SQL statements within a given time range. The SQL statement analysis provides you with a chronology of performance metrics such as execution time and lock-wait time, and the database access plans.

#### **Key Features**

This app provides the following key features:

- List of the most expensive SQL statements from the plan cache

- View of the execution and waiting behavior over time for a selected SQL statement
- Complete, formatted display of a selected SQL statement
- Access plan of a selected SQL statement

## 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.

### 5.3.1.17 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.

#### Related Information

### 5.3.1.17.1 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
BDEF	Behavior Definition	
CLAS	Class	
DCLS	ABAP Data Control Language Sources	
DDLS	Data Definition Language Source	
DEVC	Package	
DOMA	Domain	
DTEL	Data Element	
ENQU	Lock Object	
FUGR	Function Group	
FUNC	Function Module	
HTTP	HTTP Service	
INTF	Interface	
IWMO	SAP Gateway Business Suite Enable- ment - Model	Yes
IWOM	SAP Gateway: Model Metadata	Yes
IWSG	SAP Gateway: Service Groups Meta- data	Yes
IWSV	SAP Gateway Business Suite Enable- ment - Service	Yes
IWVB	SAP Gateway Business Suite Enable- ment -Vocabulary Annotation	Yes
MSAG	Message Class	
OA2S	OAuth2 Scope	Yes
PROG	Program (read only)	
SCO1	Communication: Communication Sce- nario	
SCO2	Communication: Inbound Service	
SIA1	IAM: Business Catalog	
SIA2	IAM: Restriction Type	

Object Name	Description	Generated
SIA5	IAM: Restriction Field	
SIA6	IAM: App	
SIA7	IAM: Business Catalog App Assignment	
SRVB	Service Binding	
SRVD	Service Definition	
TABL	Table Definition	
TTYP	Table Type	
XSLT	Transformation	
APIS	Whitelisting of Objects	
SRVC	Service Consumption Model	
SUSH	Authorization Default Values	
AUTH	Authorization Check Fields	
SUSO	Authorization Object	
SICF	ICF Service	Yes
SIA3	IAM: Authorization Object Extension	
DDLX	Core Data Services Metadata Extensions	
DTDC	Dictionary Dynamic Cache	
CHDO	Change Document	
NROB	Number Range Object	
ENHO	Enhancement Implementation Object	
SAJC	Application Job Catalog Entry	
SAJT	Application Job Template	
SKTD	Knowledge Transfer Document	
ENHS	ENHS Enhancement Spot	
SPRX	Proxy Object	Yes
SMBC	Maintainable Business Configuration	
SIA8	IAM: Business Role Template	
SIA9	IAM: Business Role Templ. Bus. Catalog Assignment	
UIAD	UIAD: Application Descriptors for the Fiori Launchpad Environment	
WAPA	BSP Page	Yes
SMIM	MIME Object	Yes
APLO	Application Log Object	

### i 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
TABU	Table Entries
SNUMS	Number Range Configuration

## 5.3.1.17.2 Manage Software Components

You can use this app to create, display, pull and delete software components in your ABAP environment landscape.

### Purpose

This app allows you to create a software component on a service instance and make it locally available on other instances within the landscape by pulling them.

### 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

In addition, the app supports the integration with Git.

### Supported Device Types

- Desktop
- Tablet
- Smartphone

## Related Information

[How to Display Available Software Components \[page 1072\]](#)

[How to Create Software Components \[page 1073\]](#)

[How to Pull Software Component \[page 1075\]](#)

[How to Delete Software Components \[page 1077\]](#)

### 5.3.1.17.2.1 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
<Software Component>	Name of the software component that has to be unique per service instance. The maximum length of the name is restricted to 18 characters.
<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> .
<In System>	Displays which software components have already been pulled into the service instance and are locally available there.
<Changes Available>	Shows if a new version of the software component is centrally available.

Column Name	Description
<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.
<Created 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.

## 5.3.1.17.2.2 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 *Software Component* dialog, enter a name of the software component (max. 18 characters). 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.

#### Software Component Types

Software Component Type	Description
Development	Used for standard ABAP application development.

Software Component Type	Description
Business Configuration	<p>Used to transport customizing content from one SAP Cloud Platform ABAP Environment system to another SAP Cloud Platform ABAP Environment system.</p> <p><b>i Note</b></p> <p>You can create multiple business configuration software components. You can only import one business configuration software component to a system. You can only release customizing transports to a business configuration software component</p>

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 pulled 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.
  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.

**i Note**

Note that you have to create a development package first. Afterward, 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 pulled 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 pulled.  
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 list of available software components (<Changed On> and <Created By>).

### 5.3.1.17.2.3 How to Clone Software Components

#### 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.

##### i 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 branch is automatically selected. Confirm by clicking *Clone*.
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.

#### Related Information

[How to Pull Software Component \[page 1075\]](#)

### 5.3.1.17.2.4 How to Pull Software Component

#### 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:

##### i 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 \[page 1075\]](#).

## Procedure

1. Select a software component (radio button) from the list and choose *Pull* to pull this software components. Alternatively, you can navigate to the detail page of a software component and choose *Pull* in the page header.  
In both cases, you pull the latest version of the software component. If the pull is finished and the software component is locally available on the instance, the value in the *In System* column turns to *Yes*.
2. Display the list of all pulls and checkouts 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.
3. To display or refresh the list of *Recent Actions*, choose *Go*.

### Field Description

Column Name	Description
<Software Component> (Column initially hidden)	Name of the software component that has to be unique per service instance. The maximum length of the name is restricted to 18 characters.  We recommend that you use a namespace for the software component (Z or Y). It used to check if the software component can be pulled to the service instance.
<ID> (UUID, Column initially hidden)	A unique key of the software component.
<Status>	Displays whether the pull or checkout process has been successful, is still running, or there has been an error.
<p><b>i 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.

4. You can display the pull details by choosing the entry from the list.  
The *Execution Log* and *Transport Log* are displayed as a table. You can find the new tables when navigating to the detail page of every pull 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.

## i 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 \[page 422\]](#)) to download the logs.

### Field Description

Column Name	Description
<code>&lt;Type&gt;</code>	The type of a logline can either be "Information", "Success", "Error" or "Warning". Each level is represented with a colored criticality icon.
<code>&lt;Description&gt;</code>	Displays a single log entry message.
<code>&lt;Timestamp&gt;</code>	Date and time when the entry was logged.
<code>&lt;Index (Initially hidden)&gt;</code>	The index column is initially hidden but can be enabled when clicking the <i>Settings</i> button in the table toolbar.

## Related Information

[How to Clone Software Components \[page 1075\]](#)

## 5.3.1.17.2.5 How to Delete Software Components

### Context

You want to delete a software component in order to:

- get rid of no longer used software components.
- get rid of software components that have been created by mistake (e.g. a typo in the software component name).

For every delete activity, the user is documented in a log.

## Procedure

To delete a software component, perform the following steps:

1. On the [Manage Software Components](#) screen, select a software component you want to delete.  
Alternatively, can delete a software component from the details page.
2. Choose the [Delete](#) icon in the control panel.
3. You will be prompted to confirm the action.

### Caution

Please note that this action cannot be undone.

## Result

The selected software component is deleted centrally.

You can no longer pull the software component to a service instance.

If this software component has already been pulled to a service instance, objects are not deleted in the ABAP instances but the status of the SW component and all belonging objects is changed to [read-only](#). You are 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 SW component.

## Restoring Deleted Software Components

### Note

Currently you cannot restore software components that have been deleted.

## 5.3.1.17.2.6 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 'master'. It is created when releasing a transport request for this software component using ABAP Development Tools. In the initial (development) system, the 'master' branch will not be visible until you've released a transport request. If you pull the software component on your test or production system afterwards, you'll see all branches even before you've released an ABAP transport request.
3. Note that you need to pull the repository once before you can check out a branch. Pull the repository by clicking *Pull*.
4. To create a new branch, select the branch that you want to create it from and click *+*. Choose a name for your new branch and confirm with *Create*.

### i Note

A new branch initially has the same state as the branch it was created from.

5. Refresh the page. Your new branch has now been created and added to the table.

### i Note

It is currently not possible to delete branches.

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.

### i Note

It is currently not planned to support the merging of branches.

## 5.4 Administration and Operations in the Kyma Environment

The administrators of the Kyma environment take care of setting it up and make sure it is ready for developers to work with.

### [Roles in the Kyma Environment \[page 1080\]](#)

Kyma uses roles and groups to manage access within the cluster. Every cluster comes with two predefined roles which give the assigned users a different level of permissions suitable for different purposes.

### [Provision Kyma Environment \[page 1083\]](#)

Set up a Kubernetes cluster with the project "Kyma" and use it to extend the functionality of your SAP solutions.

### [Kyma Environment Backup \[page 1083\]](#)

The user load on a cluster typically consists of various Kubernetes objects and volumes. Kyma environment relies on the managed Kubernetes cluster for periodic backups of Kubernetes objects.

## 5.4.1 Roles in the Kyma Environment

Kyma uses roles and groups to manage access within the cluster. Every cluster comes with two predefined roles which give the assigned users a different level of permissions suitable for different purposes.

The roles predefined for the Kyma runtime are:

Role	Description
<b>KymaRuntimeNamespaceAdmin</b>	The role which gives access to all Kubernetes and Kyma resources and components with administrative rights, except for the write access to <a href="#">AddonsConfigurations</a> and <a href="#">Kyma system Namespaces</a> . KymaRuntimeNamespaceAdmins can create their own Namespaces and assign the KymaRuntimeNamespaceDeveloper role to SAP Cloud Platform users.
<b>KymaRuntimeNamespaceDeveloper</b>	The role created for developers who build implementations using Kyma. It allows you to list and edit Kubernetes and Kyma-specific resources scoped to specific Namespaces.

## Related Information

[Assign Admin Role in the Kyma Environment \[page 1081\]](#)

[Assign Developer Role in the Kyma Environment \[page 1082\]](#)

## 5.4.1.1 Assign Admin Role in the Kyma Environment

As a subaccount admin, you can assign the `KymaRuntimeNamespaceAdmin` role to other SAP Cloud Platform users using SAP Cloud Platform cockpit.

### Procedure

1. Log in to SAP Cloud Platform cockpit and choose the subaccount you want to assign roles to.
2. In the left navigation panel, select the `Security` tab and choose `Role Collections`.
3. Click the `+` button and provide a name for your role collection. Enter alphanumeric characters only. For example, you can name it `KymaRuntimeNamespaceAdmin`.
4. Navigate to your role collection and click it to enter a new panel.
5. Click `Edit`.
6. Fill in fields:
  - a. In the `Roles` section, click `+` and select the `KymaRuntimeNamespaceAdmin` role from the drop-down list. The `Role Template` and `Application Identifier` fields will be populated automatically.

#### i Note

You can add more than one role to your Role Collection by repeating Step 6a.

- b. In the `Users` section, click `+` and provide the user ID. You can provide your email, SAP ID number, or both. In each case, choose `SAP ID Service` as the Identity Provider.

#### i Note

As a `KymaRuntimeNamespaceAdmin` you can only assign Kyma developer roles. To be

7. Save your configuration.

The steps listed below are **an alternative** to Step 6b.

8. In the left navigation panel, select the `Trust Configuration` tab. The `SAP ID Service` configuration is already available for you.
9. Click the entry, provide the user's email address, and click `Show Assignments`. This shows you all the role collections assigned to the user.
10. Click the `Assign Role Collection` button and select the `KymaRuntimeNamespaceAdmin` role collection from the drop-down list.

## 5.4.1.2 Assign Developer Role in the Kyma Environment

As the KymaRuntimeNamespaceAdmin, you can assign the KymaRuntimeNamespaceDeveloper role to SAP Cloud Platform users using SAP Cloud Platform cockpit.

### Procedure

1. Log in to SAP Cloud Platform cockpit and choose the subaccount you want to assign roles to.
2. In the left navigation panel, select the *Security* tab and choose *Role Collections*.
3. Click the *Add Role Collection* button and provide a name for your role collection. For example, you can name it *KymaRuntimeNamespaceDeveloper*.
4. Navigate to your role collection and click it to enter a new panel.
5. Click *Edit* to enter the edit mode.
6. Fill in all the necessary fields:
  - a. In the *Roles* section, click *+* and select the *KymaRuntimeNamespaceDeveloper* role from the drop-down list. The *Role Template* and *Application Identifier* fields will be populated automatically with relevant values.
  - b. In the *Users* section, click *+* and provide the user ID. You can provide your email, SAP ID number, or both. In each case, choose *SAP ID Service* as the Identity Provider.
7. Save your configuration.

To proceed with the next steps, log in to your Kyma environment.

8. In your Kyma Runtime, select or create the Namespace you want to assign roles to.
9. In the left navigation panel, click the *Permissions* tab to create a role binding to your role collection. To do so, click the *+ Create Binding* button. In the pop-up window, choose the *User Group* option and fill in the required fields:
  - a. As *User Group*, insert *runtimeDeveloper*.
  - b. As *Kind*, choose *ClusterRole*.
  - c. As *Role*, choose *kyma-developer*.

To proceed with the next steps, go back to SAP Cloud Platform cockpit.

10. In SAP CP cockpit, navigate to the left navigation panel and select the *Trust Configuration* tab.
11. Choose *SAP ID Service*, provide user's email address, and click *Show Assignments*. This shows you all the role collections assigned to the user.
12. Click the *Assign Role Collection* button and select the *KymaRuntimeNamespaceDeveloper* role collection from the drop-down list.

## 5.4.2 Provision Kyma Environment

Set up a Kubernetes cluster with the project "Kyma" and use it to extend the functionality of your SAP solutions.

### Procedure

1. Navigate to [Subaccounts](#).

 **Caution**

If you created a subaccount before April 23rd, 2020, you won't be able to enable the Kyma environment due to XSUAA issuer configuration issue. To fix it, change the issuer of a subaccount following the instructions in the [Rotate Signing Keys of Access Tokens \[page 975\]](#) document.

2. Access the given subaccount and select *Enable Kyma* in the *Kyma Environment* section.
3. When the dialog box opens, provide the required details and choose *Create*:
  - Cluster Name
  - Description
  - Provider
  - Region

### Results

The Kubernetes cluster with the project "Kyma" is created in the background. When it is ready, you will see the link to the Kyma Console as the [Console URL](#).

 **Remember**

This subaccount will have access to the Kyma Console only after you assign it to a proper role in the SAP Cloud Platform cockpit.

## 5.4.3 Kyma Environment Backup

The user load on a cluster typically consists of various Kubernetes objects and volumes. Kyma environment relies on the managed Kubernetes cluster for periodic backups of Kubernetes objects.

The process is automated, so it does not require any manual steps on your side. For example, Gardener uses etcd as the Kubernetes' backing store for all cluster data. This means 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 five minutes. In case the etcd database experiences any problems, Gardener automatically restores the Kubernetes cluster using the latest snapshot.

**i Note**

Volumes are not part of the backup. This means, you need to take care of creating periodic snapshots of volumes on Kubernetes.

For detailed instructions on creating on-demand volume snapshots, see [Creating On-demand Volume Snapshots](#).

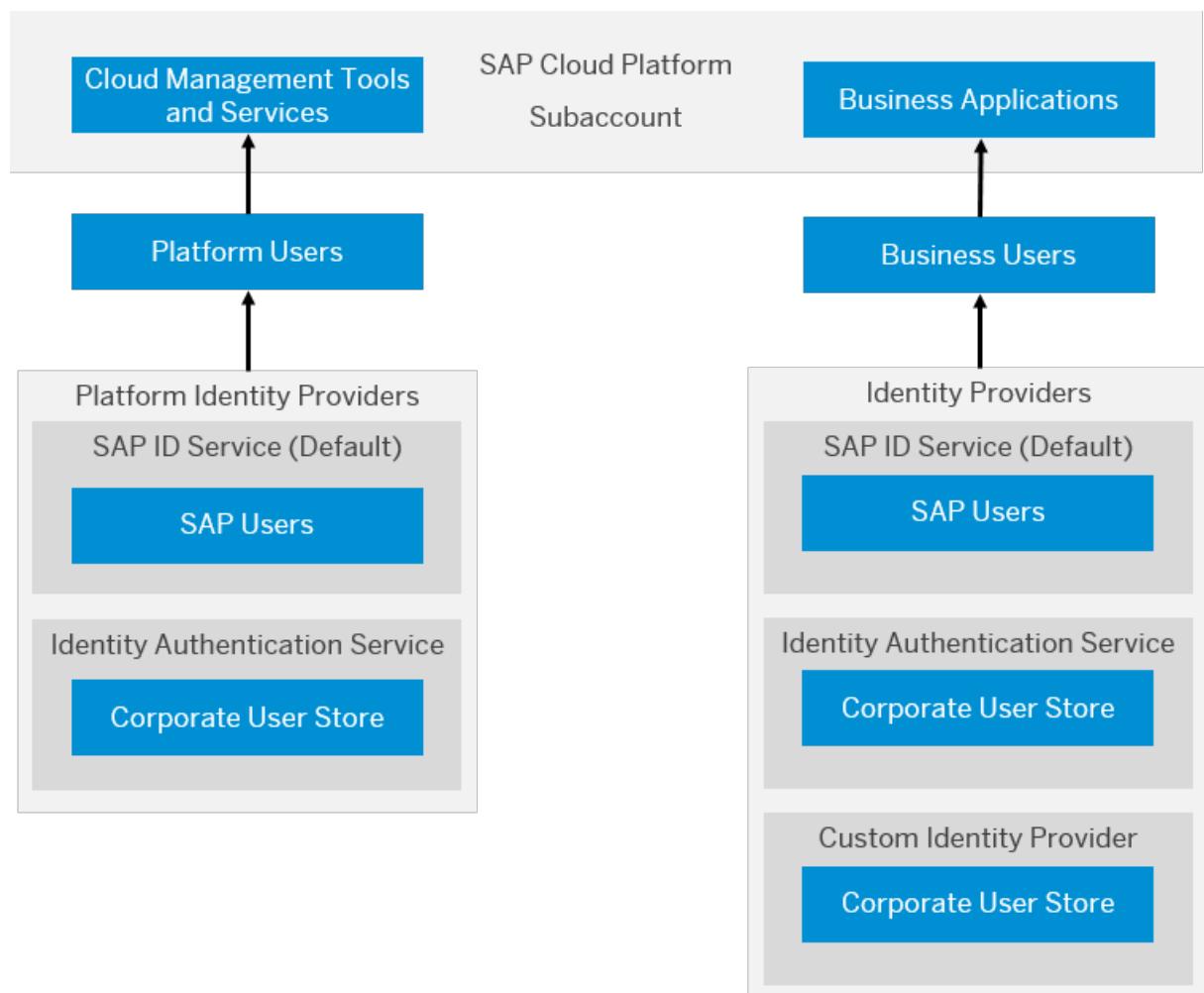
# 6 Security

Use the security features and functions of SAP Cloud Platform to support the security policies of your organization.

## User Model

SAP Cloud Platform distinguishes between **platform users** (for global accounts and subaccounts) and **business users** (for the applications).

SAP ID service is the default identity provider for both. You can configure each of your subaccounts to use a custom identity provider for its platform or business users.



See [User and Member Management](#).

## Authorizations

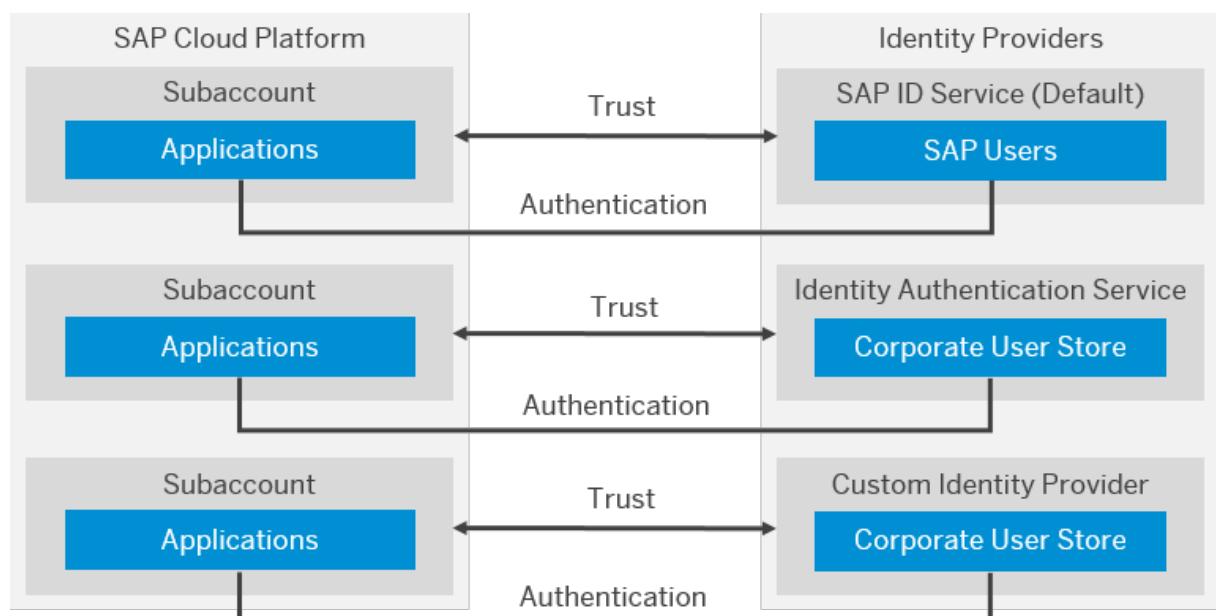
You can configure authorizations using **roles** and **role collections** for your subaccount or individual applications.

See [Security Administration: Managing Authentication and Authorization \[page 784\]](#).

## Identity Providers

Identity providers (IdP) supply the user store for your business applications or for your subaccount. You can reuse existing corporate identity management infrastructure (on-premise or on the cloud), or use the default one (SAP ID Service). You can have a different IdP for each subaccount you own, and this feature is configurable using the cockpit.

The following figure illustrates the high-level architecture of identity management in SAP Cloud Platform.



See [SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#).

## SAP ID Service

SAP ID Service is the default identity provider for both platform users and business users (in applications) at SAP Cloud Platform.

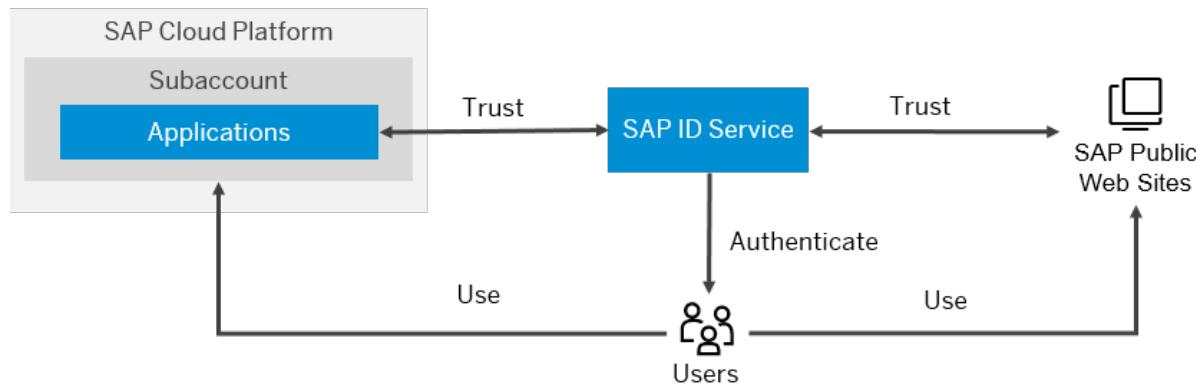
SAP ID service is the place where you register to get initial access to SAP Cloud Platform. If you are a new user, you can use the self-service registration option at the [SAP Web site](#) or [SAP ID Service](#). SAP ID Service manages the users of official SAP sites, including the SAP developer and partner community. If you already have such a user, then you are already registered with SAP ID Service.

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.

You can use SAP ID Service as a pre-configured user store in your starter scenarios, or for testing. Optionally, you can configure role assignments to SAP ID users.

See [SAP ID Service \[page 786\]](#).



## Identity Authentication Service

SAP Cloud Platform Identity Authentication service provides authentication and single sign-on in the cloud. You can use it as a corporate identity provider for your business applications. You need to configure your subaccount for trust with Identity Authentication service. See [SAP Cloud Platform Identity Authentication](#).

## Transport Layer Security (TLS) Connectivity Support

SAP Cloud Platform uses encrypted communication channels based on HTTPS/TLS. As of 30 June 2020, SAP Cloud Platform discontinues the support of TLS protocol versions 1.0 and 1.1 in all regions, and will support only TLS version 1.2 or higher.

Make sure you use HTTP clients (such as Web browsers) that support TLS version 1.2 or higher for connecting to SAP Cloud Platform.

## 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 \[page 988\]](#).

## Credential Store

SAP Cloud Platform Credential Store provides a repository for passwords and keys for applications that are running on SAP Cloud Platform. 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 Cloud Platform Credential Store](#).

## Related Information

[SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#)

[Audit Logging in the Cloud Foundry Environment \[page 988\]](#)

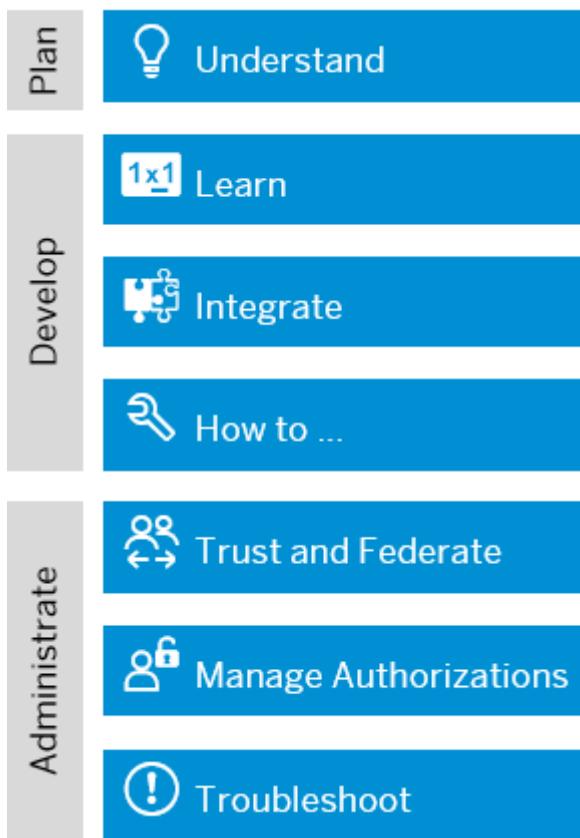
[Principal Propagation \[page 1136\]](#)

[Data Protection and Privacy \[page 1151\]](#)

[Security in the Kyma Environment \[page 1158\]](#)

## 6.1 SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment

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 cockpit.



- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im1 \[page 1089\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im2 \[page 1090\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im3 \[page 1091\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im7 \[page 1094\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im4 \[page 1091\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im5 \[page 1093\]](#)
- [#unique\\_235/unique\\_235\\_Connect\\_42\\_subsection-im6 \[page 1094\]](#)

Hold your pointer over a box for a description. Select a box to display more information.

## Authorization and Trust Management Overview

Get a high-level overview of the concepts which underpin the Authorization and Trust Management service for SAP Cloud Platform in the Cloud Foundry environment.

For more information, see [Concepts of Authorization and Trust Management \[page 1125\]](#).

## Terms and Definitions (Glossary)

Get to know important terms for understanding Authorization and Trust Management in Cloud Foundry.

For more information, see [Terms and Definitions \[page 1128\]](#).

## Tutorials for Authorization and Trust Management

Follow the tutorials below to get familiar with Authorization and Trust Management in the Cloud Foundry environment of SAP Cloud Platform.

### Tutorials for Authorization and Trust

#### Management in the Cloud Foundry

##### environment

##### Language / Framework

##### Link

Learn how to secure a basic single-tenant Node.js application with the Authorization and Trust Management service. 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 with the Authorization and Trust Management service. 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 Cloud Platform with the Authorization and Trust Management service (XSUAA) using spring-xsuaa and Spring security. Furthermore, learn how to test the secured application using the <code>java-security-test</code> 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>
Learn how to secure microservices in SAP Cloud Platform with the Authorization and Trust Management service (XSUAA). This sample provides J2EE Configuration using web.xml and uses the SAP Java Buildpack.	J2EE, SAP Java Buildpack	<a href="#">GitHub</a>
Learn how to use <code>java-security</code> 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.	Java	<a href="#">GitHub</a>

## Tutorials for Authorization and Trust

### Management in the Cloud Foundry

environment

Language / Framework

Link

Learn in this reference application how the Authorization and Trust Management service fits into a complete architecture of microservices that interact with each other.	Java	<a href="#">GitHub</a>
--	------	------------------------

## Principal Propagation

Exchange user ID information between systems or environments in SAP Cloud Platform.

### In This Section

- [Principal Propagation from the Cloud Foundry to the Neo Environment \[page 1143\]](#)
- [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 1136\]](#)

### 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

When setting up accounts you need to assign users. While we provide you with your first users to get you started, your organization has its own user bases which you want to integrate.

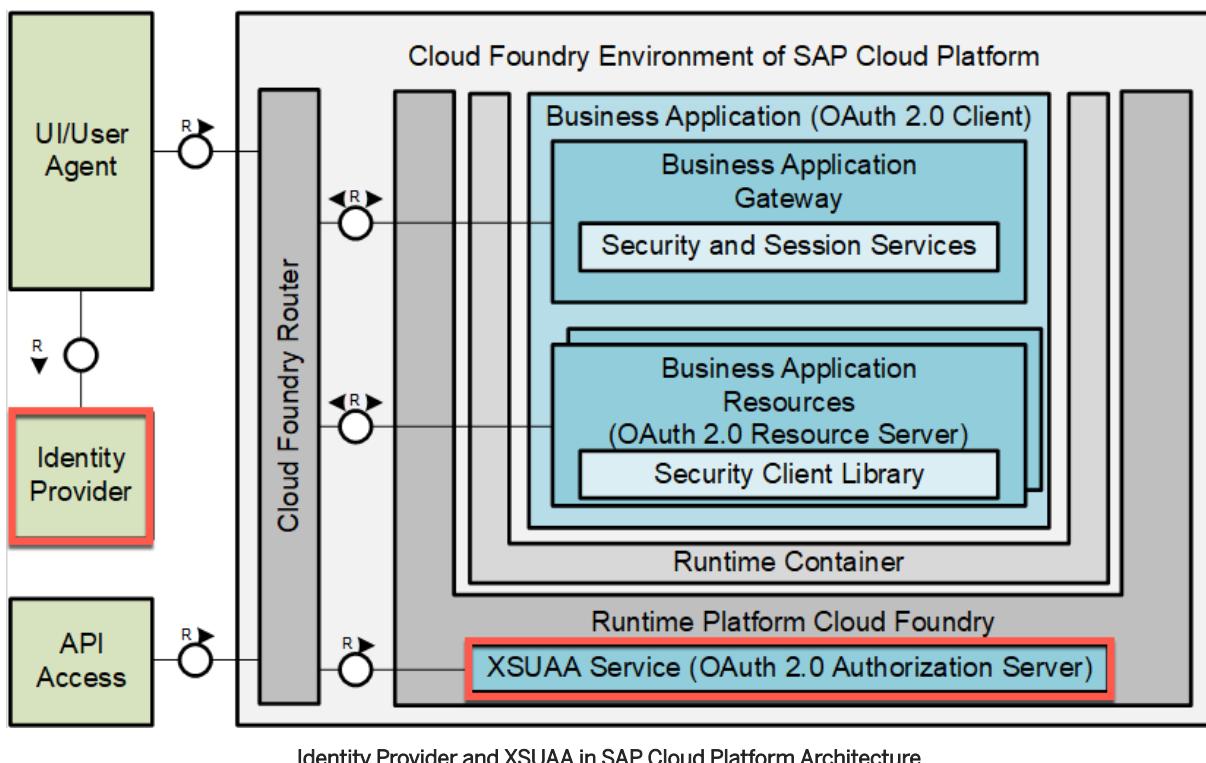
SAP Cloud Platform supports identity federation, a concept of linking and reusing digital identities of a user base across loosely coupled systems. Identity federation frees applications on SAP Cloud Platform from the need to obtain and store the credentials of users to can authenticate them. Instead, 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. To enable communication between SAP Cloud Platform 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 SAP Cloud Platform Identity Authentication service as a hub, especially if your business users are stored in multiple corporate identity providers.

For this scenario, connect Identity Authentication as single custom identity provider to SAP Cloud Platform. Next use Identity Authentication to integrate your corporate identity providers.

For more information, see [Corporate Identity Providers](#) and [Configure Conditional Authentication for an Application](#) in [What Is Identity Authentication](#) and [SAP Cloud Platform Identity Authentication service](#)



Identity Authentication is a multitenancy enabled identity management service for all applications powered by SAP Cloud Platform and optionally on-premise applications. The service provides capabilities for authentication, single sign-on, user provisioning, and on-premise integration as well as self-services like registration or password reset — for both the employees and the partners and customers of your organization. For administrators, the service offers features for user lifecycle management and reporting capabilities in the administration console.

SAP Cloud Platform has its own Identity Authentication tenant, SAP ID Service. SAP ID Service is the default identity provider of SAP Cloud Platform and where you register to get initial access to SAP Cloud Platform. Trust to SAP ID service is preconfigured by default.

We recommend that you request your own Identity Authentication tenant, but you can also use any other identity provider, which supports the SAML 2.0 protocol. To establish trust with your identity provider, perform one of the following procedures.

- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#)
- [Manually Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service \[page 794\]](#)
- [Establish Trust and Federation with UAA Using Any SAML Identity Provider \[page 798\]](#)
- [Establish Trust and Federation for Cloud Foundry Platform Users for Custom Identity Providers \[Feature Set A\] \[page 801\]](#)
- [Default Identity Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#)

#### i 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 Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#).

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 [Assign Role Collections \[page 829\]](#).

The identity provider hosts the business users, who belong to user groups. It's efficient to use federation by assigning role collections to one or more user groups. The role collection contains all the authorizations that are necessary for this user group. This method saves time when you add a new business user. Simply add the users to the respective user groups and the new business users automatically get all the authorizations that are included in the role collection.

## Administration: Managing Authentication and Authorization

In the Cloud Foundry environment, application developers create and deploy application-based authorization artifacts for business users. Administrators use this information to assign roles, build role collections, and assign these collections to business users or user groups. In this way, they control the users' permissions.

Setting Up Authorization Artifacts (Administrators)

Step	Task	User Role	Tool
1	Use an existing role or create a new one using role templates  <a href="#">Add Roles to Role Collections on the Application Level [page 826]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
2	Create a role collection and assign roles to it  <a href="#">Maintain Role Collections [page 828]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform
3	Assign the role collections to users  <a href="#">Managing Users and Their Authorizations Using the sapcp CLI [page 873]</a> or <a href="#">Assign Role Collections [page 829]</a>	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit  Command line interface for SAP Cloud Platform

Step	Task	User Role	Tool
4	(If you do not use SAP ID Service) Assign the role collections to SAML 2.0 user groups (cloud management tools feature set A regions)	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit
	<a href="#">Map Role Collections to User Groups [page 831]</a>		
5	Assign the role collection to the business users provided by an SAML 2.0 identity provider (cloud management tools feature set A)	Administrator of the Cloud Foundry environment	SAP Cloud Platform cockpit
	<a href="#">Directly Assign Role Collections to Users [page 830]</a>		

## Troubleshooting

This section provides information on troubleshooting-related activities for Authorization and Trust Management 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.

Our troubleshooting information can be found in our Guided Answers [Troubleshooting for Authorization and Trust Management in the Cloud Foundry Environment](#). Guided Answers are a specialized format to guide you step by step through troubleshooting topics. 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](#)
- [Cannot Add Role Templates to Predefined Role Collections](#)

## Developing Security Artifacts

Developers store authorization information as design-time role templates in the security descriptor file xs-security.json. Using the cockpit, administrators of the environment assign the authorizations to business users.

- [Maintenance of Application Security \[page 251\]](#)
  - [Authorization Concept for Users of an Application \[page 252\]](#)
  - [Scopes for Functional Authorization Checks \[page 252\]](#)

- [The Application Security Descriptor \[page 253\]](#)
- [Application Security Descriptor Configuration Syntax \[page 256\]](#)
- [Quick Start: Create Role Collections \(with Predefined Roles\) \[page 271\]](#)
- [Set Up Security Artifacts \[page 273\]](#)
  - [Specify the Security Descriptor Containing the Functional Authorization Scopes for Your Application \[page 275\]](#)
  - [Create Role Templates for the Application \[page 277\]](#)
  - [Create Multiple Tenants for an Application \[page 277\]](#)
  - [Create a Service Instance from the xsuaa Service \[page 278\]](#)
  - [Bind the xsuaa Service Instance to the Application \[page 280\]](#)
  - [Granting Scope Access to Another Application \[page 282\]](#)
- [Manage Authentication and Authorization Using APIs \[page 911\]](#)

## General

The Cloud Foundry environment provides platform security functions such as business user authentication, authentication of applications, authorization management, trust management, and other security functions. It enables Cloud Foundry administrators to manage authorizations and trust. Developers design authorization information and deploy this information in the Cloud Foundry environment.

## Authorizations

Applications and their users require diverse authorizations so that they can integrate seamlessly into SAP Cloud Platform. Developers configure these authorizations on the level of the application descriptor files so that security artifacts are available in the cockpit. Administrators use the security artifacts to build roles and aggregate them into role collections (sets of authorizations that are suitable for distinct user groups).

Security artifacts enable applications to communicate with other applications, for example, making or receiving calls.

## Trust Management

In SAP Cloud Platform, identity providers provide the users. You can have a different identity provider for each subaccount you own. Using the cockpit, administrators establish the trust relationship between external identity providers and the subaccounts.

### i Note

Before you start, make yourself familiar with the sections about authentication and authorization of the SAP Cloud Platform Planning and Lifecycle-Management Guide. You find the security and compliance model in the related link.

## Related Information

[What Is Authorization and Trust Management? \[page 1096\]](#)

[Security Administration: Managing Authentication and Authorization \[page 784\]](#)

[Developing Security Artifacts \[page 251\]](#)

[Setting Up Your Security and Compliance Model](#)

[Cloud Management Tools — Feature Set Overview](#)

### 6.1.1 What Is Authorization and Trust Management?

Manage application authorizations and connections to identity providers.

The SAP Cloud Platform Authorization and Trust Management service lets you manage user authorizations and trust to identity providers. Identity providers are the user base for applications. You can use an identity authentication 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 groups and role collections for large-scale cloud scenarios.

Get a high-level overview of the concepts which underpin the Authorization and Trust Management service for SAP Cloud Platform in the Cloud Foundry environment.

## Environment

This service runs in the Cloud Foundry environment Neo and Cloud Foundry environments.

## Features

<b>Use your corporate or a default IdP</b>	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.
--	---

<b>Enable role-based access to applications</b>	Enable different privileges to users accessing your applications based on roles.
---	--

For more information, see [Concepts of Authorization and Trust Management \[page 1125\]](#).

Get to know important terms for understanding Authorization and Trust Management in Cloud Foundry.

For more information, see [Terms and Definitions \[page 1128\]](#).

## Identity Federation

Identity federation is the concept of linking and reusing electronic identities of a user across multiple identity providers. This frees an application from the obligation to obtain and store users' credentials for authentication. Instead, the application reuses an identity provider that is already storing users' electronic identities for authentication, provided that the application trusts this identity provider.

This makes it possible to decouple and centralize authentication and authorization functionality. Several major protocols have been developed to support the concept of identity federation:

- SAML 2.0
- OAuth 2.0

### Authentication Using SAML 2.0

Security Assertion Markup Language (SAML 2.0) is an open standard based on XML for exchanging authentication and authorization data of a principal (user) between an identity provider (IdP) and a service provider (SP). The data is exchanged using messages called *bearer assertions*. A bearer is any party in possession of the assertion. The integrity of the assertion is protected by XML encryption and an XML signature. SAML addresses the requirement of web browser single sign-on across the Internet.

Authorizations are implemented through user groups, to which the user is assigned. How these user groups correlate to specific authorizations depends on the service provider and the respective implementation.

For more information about the SAML specification, see the related link.

### Authentication Using OAuth 2.0

The OAuth 2.0 authorization framework is a protocol for delegating authorizations. It is not suitable for authentication. The OAuth 2.0 specification defines four elements:

OAuth 2.0 Elements

OAuth 2.0 Element	Description
Resource owner (usually the end user)	A resource owner is capable of granting access to a protected resource.
Resource server (for example, a cloud application/microservice)	The resources are protected by hosts. You can reach them using REST endpoints. A resource server is capable of accepting and responding to protected resource requests using access tokens.
OAuth 2.0 client (for example, application router)	An application that makes protected resource requests on behalf of the resource owner and with its authorization.
Authorization server (User Account and Authentication service)	The User Account and Authentication service (UAA) issues access tokens for the client. Once the OAuth 2.0 client has been successfully authenticated by an SAML 2.0 compliant identity provider, it obtains the authorizations of the resource owner. An access token represents credentials used to access protected resources (see also the RFC 6749, OAuth 2.0 access token section in the related link).

## JSON Web Tokens

A JSON web token (JWT) (according to RFC 7519) is an open standard that defines a compact token format for securely 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 application and/or business application has a trust relationship with the authorization server. The trust is configured in the environment variable `<VCAP_SERVICES>` 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. The 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> ↗

### 6.1.1.1 Identity Federation

The Cloud Foundry environment of SAP Cloud Platform supports identity federation with identity providers. The current section provides an overview of the supported scenarios.

An identity provider provides the business users for SAP Cloud Platform. A mutual trust relationship between the identity provider and SAP Cloud Platform is required.

### Default Identity Provider

The default identity provider is SAP ID Service. It is part of SAP Cloud Platform. The trust relationship is already established.

For more information, see [Default Identity Federation with SAP ID Service in the Cloud Foundry Environment \[page 806\]](#).

## Custom SAML 2.0 Identity Provider

You have the option, to use any other identity provider. SAP Cloud Platform supports SAML 2.0 identity providers. If you want to use it, you must configure your own custom SAML 2.0 identity provider and establish trust between your SAP Cloud Platform subaccount and the identity provider.

The trust configuration consists of the following parts:

Configuring trust in a subaccount	<a href="#">Establish Trust with an SAML 2.0 Identity Provider in a Sub-account [page 795]</a>
Configuring trust in an SAML 2.0 identity provider	<a href="#">Register SAP Cloud Platform Subaccount in the SAML 2.0 Identity Provider [page 796]</a>
<a href="#">Trust and Federation with Identity Providers [page 789]</a>	

### 6.1.1.2 Access Management in the Cloud Foundry Environment

The Cloud Foundry environment extends SAP Cloud Platform. It provides platform security functions such as granting access to applications for business systems or business users, managing authorizations, and other security functions.

Applications contain content (for example, Web content, micro services), which is deployed to different containers. Business users use a user interface or a user agent to access the content of the applications whereas business systems use APIs to do so. The applications are using OAuth 2.0 to authenticate against the User Account and Authentication service.

## OAuth 2.0

The Cloud Foundry environment uses OAuth 2.0 as authentication method and access tokens. An SAML identity provider stores the business users. The applications (for example, Java or Node.js) are deployed in the runtime container. There are multiple ways of accessing the applications in the runtime container:

- Web access  
Business users access the runtime container over the Web, using a browser or a browser-based user interface.
- API access  
Using APIs, business systems directly access the runtime container of the Cloud Foundry environment.

For more information, see the related links.

## Web Access

Access to the static Web content requires user authentication and the appropriate authorization.

Applications authenticate using OAuth 2.0. They use the User Account and Authentication (UAA) service as OAuth 2.0 authorization server, the application router as OAuth 2.0 client, and application logic running in Node.js and Java backend services as OAuth 2.0 resource server.

The authentication process is triggered by the application router component, which is configured in the design-time artifact `xs-app.json`, if required. Authorization restricts the access to resources based on defined user permissions. Resources in the context of applications are services provided by a container (for example, an OData Web service) or SAP HANA database artifacts.

## API Access

A business system uses APIs to directly access the resources in the runtime container. After having authenticated at the UAA, which acts as OAuth 2.0 authorization server, the business system gets the appropriate access token. It enables the APIs to make calls into the applications of the runtime containers.

## User Account and Authentication Service

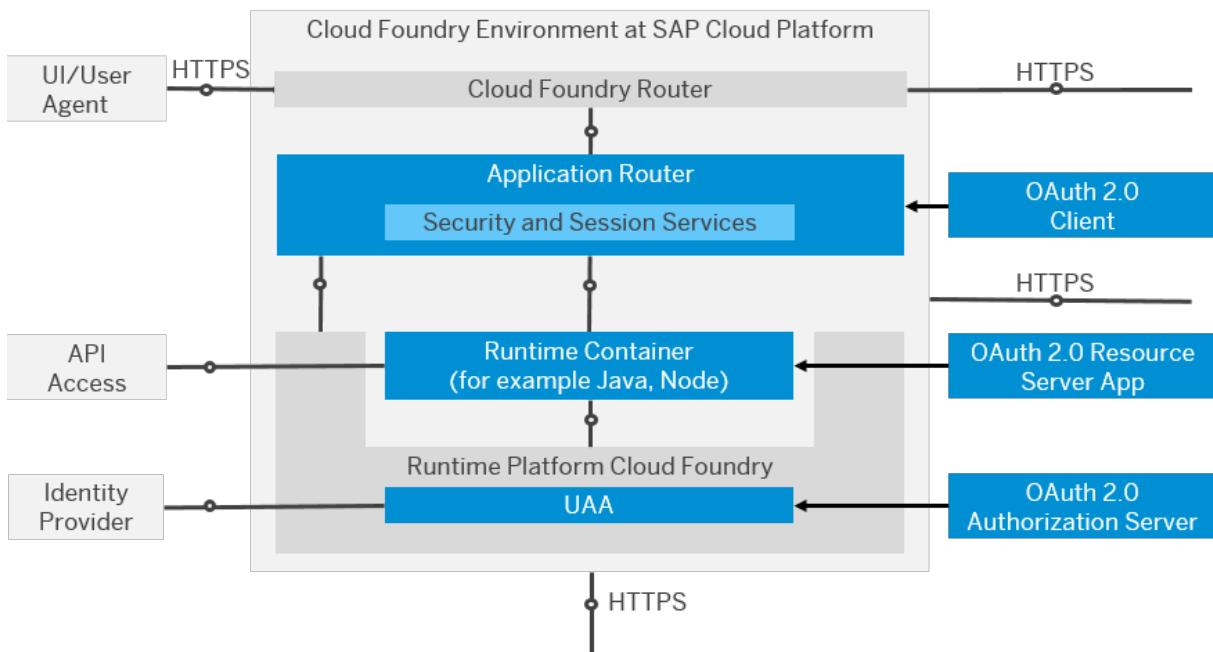
You want to integrate an application in the Cloud Foundry environment. This means that the Cloud Foundry environment needs to know your application. Using the User Account and Authentication (UAA) service, you initially integrate your application into the platform environment of SAP Cloud Platform. The application needs to authenticate against the User Account and Authentication service. The authentication concept of the Cloud Foundry environment is OAuth 2.0.

In this context, the UAA acts as OAuth 2.0 authorization server. The application router itself is the OAuth 2.0 client. To integrate the application into authentication, you must create a service instance of the `xsuaa` service and bind it to the application router and the application containers. From the OAuth 2.0 perspective, the containers (for example the Node.js container) are OAuth 2.0 resource servers. Their container security API validates the access tokens against the UAA.

The UAA uses OAuth 2.0 access tokens with the JSON web token (JWT) format for authenticating at the containers, the OAuth 2.0 resource servers.

## Runtime Containers

Static Web content is deployed into the application router, application logic, for example, into Node.js and Java runtime containers.



## Related Information

[Web Access in the Cloud Foundry Environment \[page 1104\]](#)

[API Access in the Cloud Foundry Environment \[page 1112\]](#)

[Developing Security Artifacts \[page 251\]](#)

### 6.1.1.2.1 User Account and Authentication Service of the Cloud Foundry Environment

The User Account and Authentication service (UAA) is the central infrastructure component of the Cloud Foundry environment at SAP Cloud Platform for user authentication and authorization.

#### UAA Instances

The Cloud Foundry environment at SAP Cloud Platform distinguishes between two user types and manages each type in a separate UAA instance. This means the two types are completely separated:

- Platform users perform technical development, deployment, and administration tasks. They use tools like the cloud cockpit and the `cf` command line client. These users typically have authorizations for certain organizations and/or spaces, and other technical services in the Cloud Foundry environment. Apart from authentication to the platform using the `cf` command line client, usually there is no direct interaction between users and the **platform UAA**.

- Business users only use business applications deployed to SAP Cloud Platform. They do not use SAP Cloud Platform for technical development, deployment, and administration tasks. A business user is always bound to a specific tenant which holds the information about the user's authorizations. Tenants, business users, and their authorizations are managed by another UAA instance using the extended services for UAA (xsUAA). This component additionally provides a simple programming model for authentication and authorization in business applications.

This documentation refers to business users and the extended services of the UAA.

## UAA Supports OAuth 2.0

The UAA uses OAuth 2.0 for authentication of the application. In the context of the OAuth flow, the UAA provides scopes, role templates, and attributes to applications deployed in the runtime of the Cloud Foundry environment. If a business user has a certain scope, an access token and a refresh token with this scope are issued. These enable the user to run the application, while the scopes are used to define the model that describes who has the authorization to start an application that runs in the runtime of the Cloud Foundry environment at SAP Cloud Platform.

The UAA service provides a programming model for developers; it enables developers to define templates that can be used to build role and authorization models for business users of applications deployed to the runtime of the Cloud Foundry environment. In the OAuth 2.0 workflow, the UAA acts as an OAuth server.

### i 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.

## Related Information

[Cloud Foundry API Reference for User Account and Authentication Server API](#) 

[UAA, XSUAA, Platform UAA, CFUAA - What Is It All About?](#) 

## 6.1.1.2.2 The XSUAA Programming Model

The XSUAA programming model supports multiple tenants in SAP Cloud Platform. In the context of OAuth 2.0, the security APIs for the application runtime container act as resource servers and provide the security context for user authentication in an application.

OAuth 2.0 is using SAML 2.0 for authentication of the users, which are provided by identity providers.

### Multiple Tenants

SAP Cloud Platform supports multitenant applications. These applications are used by multiple tenants. With this approach, the tenants share the same code base, but they are not allowed to see the data of other tenants. The application must maintain data separation between tenants.

### Security Services in the Application Router

In the Cloud Foundry runtime, SAP Cloud Platform provides the application router as an option. It is a point-of-entry for web applications running on the Cloud Foundry environment at SAP Cloud Platform; the application router is part of the application and triggers the user authentication process in the UAA. In the OAuth 2.0 workflow for web access, the application (including the application router and any bound containers) is the OAuth 2.0 client, which initiates user authentication and authorization.

### Container Security APIs

In the context of OAuth 2.0, the security APIs for the runtime container act as resource servers. The APIs provide the security context for user authentication in an application of the Cloud Foundry environment. When the user authentication process is triggered, the container security APIs receive an `Authorization: Bearer` HTTP header that contains an OAuth access token in the JSON Web token (JWT) format from the application router.

OAuth access tokens expire after a certain period of time. After they have expired, they cannot be used for authentication anymore. The application router supports the token refresh flow according to the OAuth standard. Applications use refresh tokens to renew access tokens. For web applications, the application router initially requests an access token from the UAA. The UAA responds with an OAuth access token and an additional OAuth refresh token. When the current access token has expired, the application router uses this refresh token to get a new token from the UAA. You can change the token settings in the OAuth 2.0 client configuration of the security descriptor file (see the related link).

The access token and the refresh token contain information describing the user and any authorization scopes. These must be validated by the container security APIs using the security libraries provided by the UAA. Applications deployed in the Cloud Foundry runtime at SAP Cloud Platform can use the security API to check whether scope values have been assigned to the user or application. The container security API provides the security context for applications (for example, scopes, attributes, token information) of the application; the

JWT token initializes the security context of the application. A security API is available for the following application runtime containers:

Runtime Container	Authentication Reference
Node.js	<a href="#">Authentication for Node.js Applications [page 1118]</a>
Java API using Spring	<a href="#">Configuring Authentication for Spring Boot Applications [page 1117]</a>
Java web application using sap_java_buildpack	<a href="#">Configuring Authentication for SAP Java Buildpacks [page 1117]</a>

For more information, see the related links on authentication. They describe how you configure authentication for Node.js, Java.

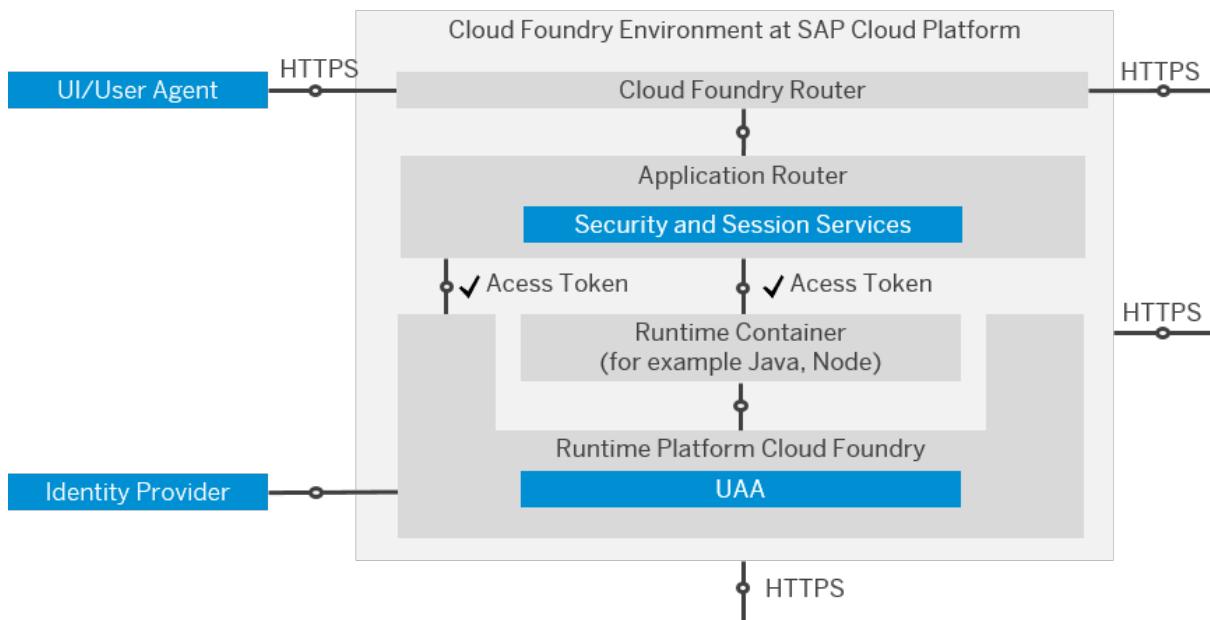
Each application's runtime container of an application must use the container security API to provide the security context. The security context is initialized with the JWT token and enables you to perform the following functions:

- Get user information
- Check an authorization scope
- List authentication attributes
- Get an access token and a refresh token

### 6.1.1.2.3 Web Access in the Cloud Foundry Environment

The Cloud Foundry environment extends SAP Cloud Platform. 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 Platform Identity Authentication service
- Any SAML 2.0 identity provider

Applications use 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.

## Related Information

[Authentication for Applications \[page 1115\]](#)

[Authorization for Applications \[page 1124\]](#)

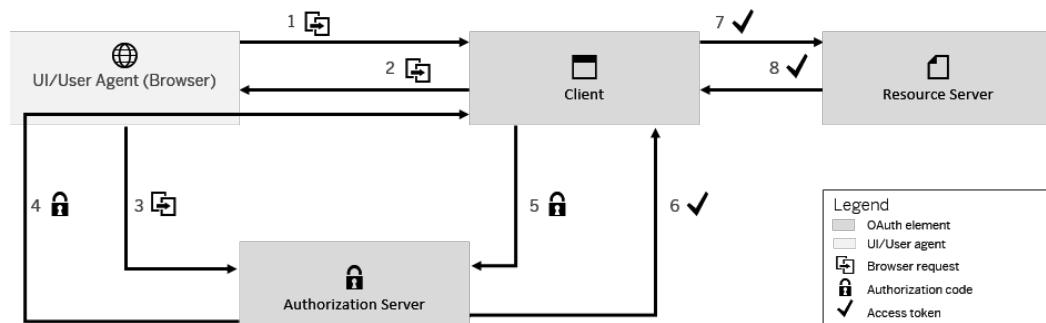
### 6.1.1.2.3.1 OAuth 2.0: Authorization Code Grant Type

In SAP Cloud Platform, the UAA uses OAuth 2.0 for authentication of the applications deployed in the runtime of the Cloud Foundry environment.

To access an application with OAuth 2.0, an OAuth 2.0 client must authenticate with an access token. The flow of the authorization code grant type is used to get an initial access token and a refresh token from an OAuth 2.0 authorization server. The OAuth 2.0 client can then use the refresh token to request new access tokens itself whenever an access token expires.

An application wants to access a resource on an OAuth 2.0 resource server using a browser-based user interface. To do this, it must have a valid access token. For more information about the OAuth 2.0 configuration, see the related link.

The following diagram illustrates how it uses the authorization code grant type to get an access token:



1. A user of a browser-based application (user agent) sends a request to the OAuth 2.0 client.
2. Since the application has no access token, the client redirects the request to the browser.
3. The application requests an authorization code at the authorization server.
4. The authorization server checks the validity of the request and grants an authorization code to the client.
5. The client receives the authorization code and requests an access token from the authorization server.
6. The authorization server issues an access token and grants it to the client.
7. The client presents the access token to request the resource on the resource server.
8. In the final step, the resource server validates the access token and allows the client to access the resource.

The authorization code grant type conforms to the RFC 6749 standard of IETF. For more information, see <http://ietf.org>.

#### → Tip

To mitigate the impact of cross-site scripting attacks, we recommend that you avoid sending access tokens to the browser.

When an application is using the application router, it shares a security session with the browser. The security session holds the access and refresh token. Requests to back-end systems include the JSON web token (if configured).

## Related Information

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

## 6.1.1.2.3.2 Domain Checks at Browser Login/Logout

When a user logs in to a web application or logs out using a browser, component `xsuaa` checks the domain in the URL of the application routes. To avoid open redirect attacks, `xsuaa` checks the URLs, which are defined in

`xs-app.json`, against a whitelist of redirect URIs in the application security descriptor file (`xs-security.json`).

A web application in the Cloud Foundry environment of SAP Cloud Platform has application routes. These are URLs that point to the web application. At runtime, component `xsuaa` checks whether the redirect URI for login and logout is correct and rejects any attempts to access incompatible redirect URIs. Component `xsuaa` checks the domains in these URLs during authentication and logout. A successful check ensures that users who log out are redirected to a URL which reflects the logout and thus makes sense for the users.

## Default Domain

Usually, the application uses the default domain in the application route URL. For this reason, enter the relevant application route.

### • Example

```
<application_name>.cfapps.eu10.hana.ondemand.com (example for the eu10 landscape)
```

### • Example

For China (Shanghai) region:

```
<application_name>.<custom_domain>.<landscape_domain>.hana.ondemand.com
```

## Custom Domain

See the SAP Note in the related link for information on how to set custom domains in the Cloud Foundry environment of SAP Cloud Platform.

As an application developer, if you use custom domains in the application routes of your web application, you must include the login and logout URLs in the `xs-security.json` of the `xsuaa` service instance. The configuration of the redirect URIs is located in the `oauth2-configuration` custom option of the security application descriptor file (`xs-security.json`). Here, you have to define the URLs in `redirect-uris` of the custom options in the OAuth 2.0 configuration. You can also use wild cards (\*) to allow multiple URLs that belong to one domain.

For more information, see the related links.

## Related Information

[Configure Redirect URLs for Browser Logout \[page 1108\]](#)

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

[Configuring Application URLs](#)

[SAP Note 2668456 on setting custom domains](#)

### 6.1.1.2.3.3 Configure Redirect URLs for Browser Logout

To avoid open redirect attacks, you want to direct users to a safe and valid URL when they log out.

#### Prerequisites

- You have 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 have deployed an application in the Cloud Foundry environment of SAP Cloud Platform. To use the application, business users log in and out. As an application developer, you want to set a redirect URI, which directs users to a specific page once they are logged out. This could be a logout page or your company's employee portal. For security reasons, there is 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 whitelist of allowed redirect URLs. You define them in the OAuth 2.0 configuration parameter 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. For more options, see [Browser Redirects Using Wildcards \[page 1110\]](#).

##### ↳ 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 (CLI).
4. Choose your subaccount.

### i Note

The Cloud Foundry command line interface 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 \[page 917\]](#)).

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` parameter.
9. Also add a URL that contains the domain of your application.

`https://<application_domain>.*/**`

### ↳ Sample Code

```

{
    "xsappname": "<application_name>",
    "tenant-mode": "dedicated",
    "description": "My Sample Application with Application Router",
    "oauth2-configuration": {
        "token-validity": 900,
        "redirect-uris": ["https://
<application_domain>.cfapps<custom_domain>.eu10<landscape_domain>.hana.ondemand.com/**"]
    },
    "autoapprove": "true"
}
}

```

### i 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
```

### i Note

If the service instance is not available, create it, bind your application to the service instance (see the related links), and restage the application.

You have implemented a valid redirect after logout, and xsuaa checks the redirect URL against the whitelist in the OAuth 2.0 configuration.

## Related Information

[Application Router \[page 77\]](#)

[Configuring Application URLs](#)

[Update a Service Instance \[page 279\]](#)

[Create a Service Instance from the xsuaa Service \[page 278\]](#)

[Bind the xsuaa Service Instance to the Application \[page 280\]](#)

[Cloud Foundry Command Line Interface](#) ↗

## 6.1.1.2.3.3.1 Browser Redirects Using Wildcards

You want to configure browser redirect URLs for multiple external web sites of your company. We recommend that you specify absolute URLs and avoid using wildcards.

### ⚠ Caution

When using wildcards in the redirect URLs, we remind you for security reasons to be aware that you open up the redirect for multiple web sites. Be aware that this increases the risk of redirecting to malicious web sites.

You want to allow multiple redirect URLs, for example in the `sap.com` domain. Use wildcards in the redirect URLs of the OAuth 2.0 configuration of `xs-security.json`. In this example, the configuration in `xs-security.json` allows external redirect URLs that contain `sap.com`.

## Configuration of `xs-app.json`

In the following example, you configure a logout page, which is a static external URL. Define the external URL in `xs-app.json`.

### • Example

#### ↳ Sample Code

```
"logout": {  
    "logoutEndpoint": "/my/logout",  
    "logoutPage": "https://support.sap.com"
```

## Configuration of `xs-security.json`

The following example allows redirects to all web sites with `sap.com` as domain. It allows secure (`https`) and (`http`) pages. Define redirect URLs with wildcards.

### • Example

#### ↳ Sample Code

```
"oauth2-configuration": {  
    "token-validity": 900,  
    "redirect-uris": ["http*://*.sap.com/**",  
                    "https:///  
<application_domain>.cfapps.eu10.hana.ondemand.com/**"]}
```

### • Example

For China (Shanghai) region:

#### ↳ Sample Code

```
"oauth2-configuration": {  
    "token-validity": 900,  
    "redirect-uris": ["http*://*.sap.com/**",  
                    "https:///  
<application_domain>.<custom_domain>.cn/**"]}
```

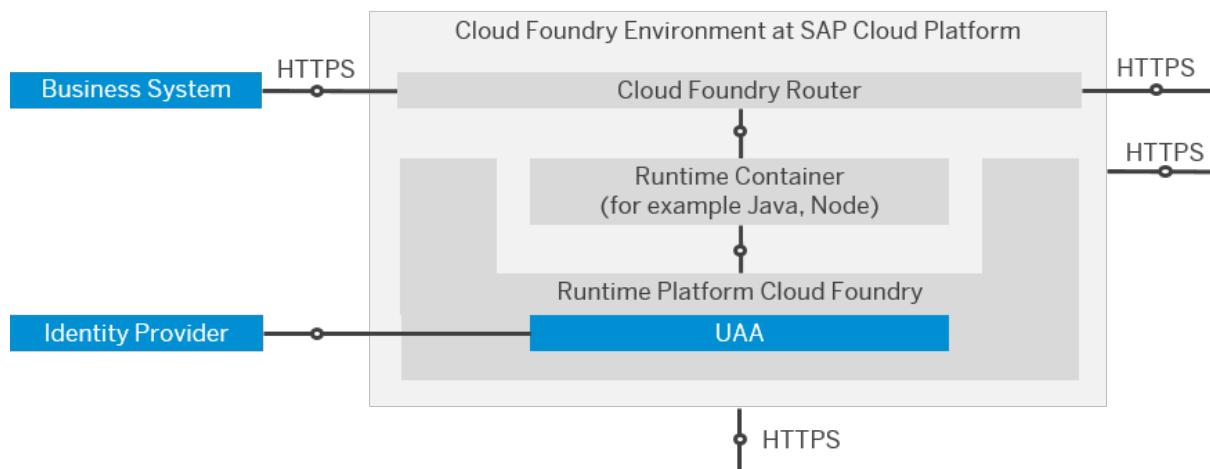
## Related Information

[Configure Redirect URLs for Browser Logout \[page 1108\]](#)

### 6.1.1.2.4 API Access in the Cloud Foundry Environment

The Cloud Foundry environment extends SAP Cloud Platform. 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.

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 SAML 2.0 identity providers:

- SAP Cloud Platform Identity Authentication Service
- Any SAML 2.0 identity provider

Business systems use OAuth 2.0 to access applications in the runtime container. The UAA acts as an OAuth authorization server and issues an 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

[Authentication for Applications \[page 1115\]](#)

[Authorization for Applications \[page 1124\]](#)

[Get API Access \[page 912\]](#)

## 6.1.1.2.4.1 Authentication for Business Users Using the SAML 2.0 Bearer Assertion Flow

A business application needs to access OAuth-protected resources in SAP Cloud Platform on behalf of a business user. It is clear that this business user expects his/her user to be propagated smoothly from one application to the next. What is used here is SAML identity federation.

In this context, the business user's user ID and attributes from a SAML assertion are exchanged for an OAuth access token. The business user can use the access token to access the OAuth-protected application that holds the resources. For more information, see the [relevant section](#). This example of the SAML 2.0 bearer assertion flow describes the case for business users who log on to a web application and want to use web-based resources in the Cloud Foundry environment of SAP Cloud Platform.

### Prerequisites

- There must be a mutual SAML trust between the business application and SAP Cloud Platform. To implement SAML trust, see the related link. We recommend that you disable the [\*Show SAML login link on login page\*](#) checkbox.

### SAML 2.0 Bearer Assertion Flow

#### i Note

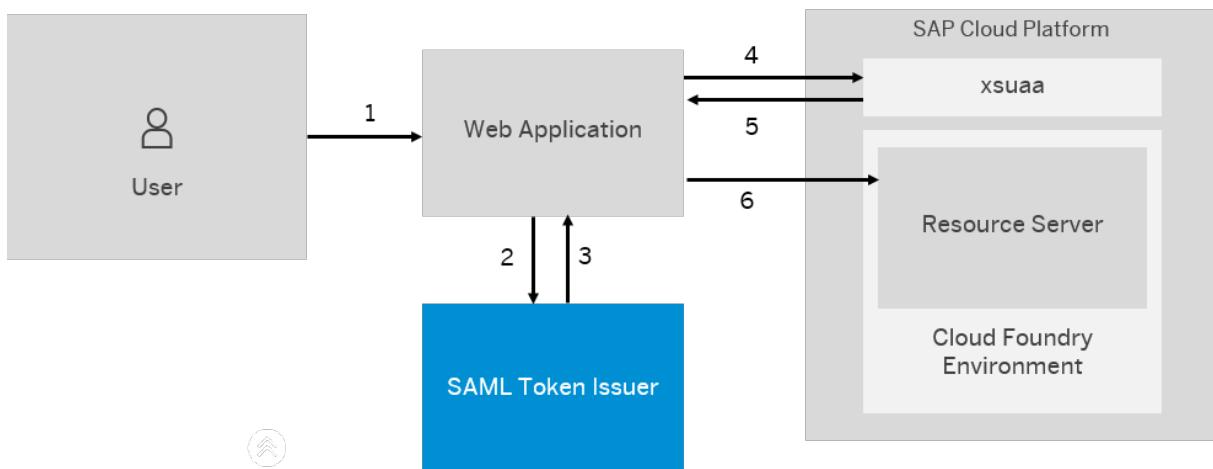
The SAML 2.0 bearer assertion can be issued by an SAML token issuer. The SAML token issuer in the example below might be a SAML 2.0 identity provider.

#### ❖ Example

The following example shows the SAML 2.0 assertion bearer flow with a business user calling from a web application to a resource server in the Cloud Foundry environment.

The SAML 2.0 assertion bearer flow has the following steps:

1. The business user accesses the web application, which uses a resource server hosted in the Cloud Foundry environment of SAP Cloud Platform.
2. The application requests a SAML bearer assertion.
3. The SAML token issuer issues the SAML bearer assertion.
4. The web application calls the `xsuaa` service and requests an OAuth access token (for details, see below).
5. The `xsuaa` service returns the OAuth access token with the received SAML bearer assertion.
6. The web application can access the resource server on behalf of the business user by providing the OAuth access token.



## OAuth Access Token

The SAML 2.0 assertion bearer flow of OAuth 2.0 is the authentication method here. Business users use the SAP Cloud Platform to log on to the web application. During logon, the business users are provided with a token.

To obtain an OAuth access token, the request with the parameters you see in the table below must be send to the `xsuaa` service. You find the exact URL in the SAML metadata.

### Example

This is an example for the eu10 landscape.

1. Obtain the SAML 2.0 metadata document for your subaccount. It is located at `https://<subdomain_name>.authentication.<landscape_domain>.hana.ondemand.com/saml/metadata`.  
For cloud management tools feature set B: For eu10, it is `https://<subdomain_name>.authentication.eu10.hana.ondemand.com/saml/metadata`.
2. Go to the end of the document and look for the value `Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"` as `AssertionConsumerService`.

### Request Parameters

Parameter	Type	Constraints	Description
<code>client_id</code>	String	Optional	OAuth client ID of the OAuth client that receives the token
<code>client_secret</code>	String	Optional	Client secret configured for the OAuth client. This is optional if passed as part of the basic authorization header

Parameter	Type	Constraints	Description
<code>grant_type</code>	String	Required	This is the type of token grant requested. Use the following value:  <code>urn:ietf:params:oauth:grant-type:saml2-bearer</code>
<code>assertion</code>	String	Required	XML based SAML 2.0 bearer assertion, which is Base64 encoded
<code>scope</code>	String	Optional	List of scopes requested for the token. Use this list if you want to reduce the number of scopes the token will have.

This token enables the web application to connect to the Cloud Foundry environment of SAP Cloud Platform and access the desired resources.

## Related Information

[SAML2 Bearer Grant in the Cloud Foundry Documentation](#) ↗

[Trust and Federation with Identity Providers \[page 789\]](#)

[Principal Propagation to OAuth-Protected Applications](#)

### 6.1.1.3 Authentication for Applications

The application router, the application containers, and the User Account and Authentication service are required for the authentication methods for user and application logon requests.

The Cloud Foundry environment at SAP Cloud Platform uses the User Account and Authentication service (UAA) and the application router to manage user logon and logoff requests. The UAA service centrally manages the issuing of tokens for propagating the identity to application containers and the SAP HANA database. Applications contain content, which is deployed to containers. Access to the deployed content requires user **authentication**. The following components are required for authentication:

- Application router  
The application router is a part of the application instance. It triggers authentication of the user at the UAA. The application instance (with containers and application router) is the OAuth 2.0 client, which handles authentication and authorizations. The application router is the single entry point for the authentication of a (business) application. It has the responsibility to serve static content, authenticate users, rewrite URLs, and make proxy requests to other micro services while propagating user information.
- Application container  
The containers, for example for Node.js or Java, act as resource servers providing the security context for authentication. The container security APIs receive an HTTP header with `Authorization: Bearer` and the JWT token from the application router. This token contains user, scope and token information and is validated by the container security APIs using the UAA. Applications can use the API to check if scope values have been assigned to the user or application.

For more information, see the related link.

- User Account and Authentication (UAA)

The User Account and Authentication component `xsuaa` provides a programming model for business applications. It is the central infrastructure component of the runtime platform for business user authentication and authorization management.

## Related Information

[Application Router \[page 77\]](#)

[Authentication for Node.js Applications \[page 1118\]](#)

[Authentication for Java Resource Servers \[page 1116\]](#)

### 6.1.1.3.1 Authentication for Java Resource Servers

The Cloud Foundry environment at SAP Cloud Platform provides Java runtimes to which you can deploy your Java applications.

Authentication for Java applications relies on a usage of the OAuth 2.0 protocol, which is based on central authentication at the UAA. The UAA vouches for the authenticated user's identity using an OAuth 2.0 access token. The current implementation uses as access token a JSON web token (JWT). This is a signed text-based token in JSON syntax. The Java application is specified in the related manifest file.

Validation of JSON web tokens is supported for the following applications:

- Spring Boot web applications
- Java web application using `sap_java_buildpack`
- Java applications using cloud security

The Java buildpacks use different authentication methods. For more information, see the related links.

During application deployment, the build pack ensures that the correct SAP Java Virtual Machine (JVM) is provided and that the appropriate data sources are bound to the corresponding application container.

#### i Note

SAP Cloud Platform makes no assumptions about which frameworks and libraries to use to implement the Java micro service.

## Related Information

[Configuring Authentication for Spring Boot Applications \[page 1117\]](#)

[Configuring Authentication for SAP Java Buildpacks \[page 1117\]](#)

[Configuring Authentication for Java Using Cloud Security \[page 1118\]](#)

## 6.1.1.3.1.1 Configuring Authentication for Spring Boot Applications

Spring Boot applications using the Spring-security libraries can integrate with the xsuaa service.

SAP Cloud Platform provides a library for integrating validation of tokens issued by the xsuaa service with Spring-security. The signature is validated using the verification key received from the xsuaa service.

Library and Source

Description	Source
The library is available on the Central maven repository.	<groupId>com.sap.cloud.security.xsuaa</groupId> <artifactId>spring-xsuaa</artifactId>
Source and further documentation	<a href="#">GitHub</a>

To authenticate requests with JSON Web tokens (JWT), see the following sample application:

[Sample application for Spring security](#)

## 6.1.1.3.1.2 Configuring Authentication for SAP Java Buildpacks

The SAP Java buildpack includes the XSUAA authentication method for J2EE applications. This method makes an offline validation of the received JWT token possible. The signature is validated using the verification key received from the service binding to the Authorization and Trust Management service (xsuaa ).

You've created the xsuaa service instance. (See the related link.)

The following table provides you with code samples for configuring authentication. SAP Cloud Platform offers an offline validation of the JWT token. The trust for this offline validation has been created by binding the xsuaa service instance to your application.

Source

Description	Source
The library is available on the Central maven repository.	<groupId>com.sap.cloud.security.xsuaa</groupId> <artifactId>api</artifactId> <provided>true</provided>
Source and further documentation	<a href="#">GitHub</a>

To enforce a check for the \$XSAPPNAME.Display scope, see the following sample application:

[Sample application for SAP Java buildpack](#)

## Related Information

[Create a Service Instance from the xsuaa Service \[page 278\]](#)

### 6.1.1.3.1.3 Configuring Authentication for Java Using Cloud Security

Java applications using the Java-security libraries can integrate with the xsuaa service.

SAP Cloud Platform provides a Java native library for integrating a validation of tokens issued by the xsuaa service. The signature is validated using the verification key received from the xsuaa service.

Source

Description	Source
The library is available on the central maven repository.	<pre>&lt;groupId&gt;com.sap.cloud.security&lt;/ groupId&gt;  &lt;artifactId&gt;java-security&lt;/artifactId&gt;</pre>
Source and further documentation	<a href="#">GitHub</a>

To authenticate requests with JSON Web tokens (JWT), see the following sample application:

[Sample application for Java security](#)

### 6.1.1.3.2 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 Cloud Platform.

SAP Cloud Platform 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 Cloud Platform 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://npm.sap.com>
- 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 Cloud Platform.

## **@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>"}, ... ]
```

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`).

#### i 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.

#### i 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	<code>@sap:registry = "https://npm.sap.com"</code>
Default NPM registry	<code>registry = "http://registry.npmjs.org/"</code>

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> 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> 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.

### 6.1.1.3.3 Cross-Application Authentication

In the Cloud Foundry environment of SAP Cloud Platform, an application makes API calls to other applications.

Depending on the use case, it may also be necessary for an application from an external system to make API calls into applications running in the Cloud Foundry environment.

#### Cross-Application Calls to Applications Inside the Cloud Foundry Environment

The User Account and Authentication (UAA) service is the central authentication service in the Cloud Foundry environment of SAP Cloud Platform. The UAA service and the application router manage user logon and logoff requests. The UAA service centrally manages the issuing of JSON web tokens for propagating the identity to the applications in the Cloud Foundry environment.

The interaction is browser based. Users are propagated in the following ways:

- Using an SAML 2.0 identity provider

#### 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 SAML 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 an SAML 2.0 bearer assertion or client credentials (JSON web tokens) to propagate named users

## Related Information

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

[Principal Propagation \[page 1136\]](#)

### 6.1.1.3.4 Validation and Revocation of Access Tokens

To confirm that access tokens were truly issued by the SAP Cloud Platform 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 Cloud Platform 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 Cloud Platform 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 Cloud Platform 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 Cloud Platform 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 \[page 975\]](#)

[Application Security Descriptor Configuration Syntax \[page 256\]](#)

<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> ↗

[Authentication for Applications \[page 1115\]](#)

[XSUAA authentication and authorization service integration libraries and samples for authenticating users and services on GitHub](#) ↗

### 6.1.1.4 Authorization for Applications

Authorization restricts access to resources and services based on defined user permissions.

Applications of the Cloud Foundry environment at SAP Cloud Platform contain content that is deployed to backing services. Access to the content requires not only user authentication but also the appropriate **authorization**.

The access-control process controlled by the authorization policy can be divided into the following two phases:

- Authorization  
Defined in the deployment security descriptor (`xs-security.json`), where access is authorized
- Policy enforcement  
The general rules by which requests for access to resources are either approved or disapproved.

Access enforcement is based on user identity and performed in the following distinct application components:

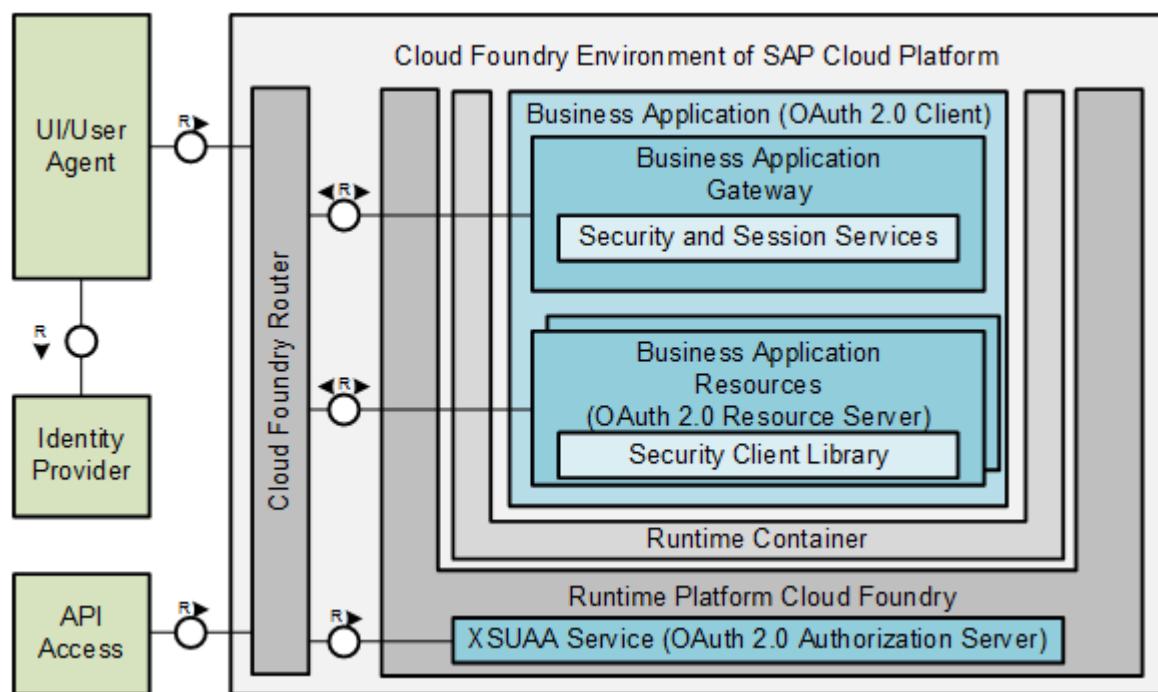
- Application router  
After successful authentication at the application router, the application router starts an authorization check (based on scopes).
- Application containers  
For authentication purposes, containers, for example the node.js and Java containers, receive an HTTP header `Authorization: Bearer <JWT token>` from the application router; the token contains details of the user and scope. This token **must** be validated using the Container Security API. The validation checks are based on scopes and attributes defined in the deployment security descriptor (`xs-security.json`).

## 6.1.2 Concepts of Authorization and Trust Management

A high-level overview of the concepts, which underpin the Authorization and Trust Management service for SAP Cloud Platform in the Cloud Foundry environment.

### Overview

The components and their interactions are depicted in the block-diagram below. It shows a high-level overview of those components, which comprise an SAP Cloud Platform business web application and how these 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).



### Application, Microservice, and App

The Cloud Foundry environment of SAP Cloud Platform is a runtime platform for business web applications. These are referred to as "Applications" for the further discussion. An SAP Cloud Platform 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. That 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 Cloud Platform, microservices are referred to as "Apps".

## Application Architecture

The application consists of a distinct gateway app with at least one or more resource app(s). 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 app(s) 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.

## Service, Service Broker, and Service Instance

A service is an app including 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 Cloud Platform, and acting on requests from the platform for provisioning, binding, unbinding, and de-provisioning.

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 app(s). The service binding contains information about the service (for example, URL, credentials, etc.), which the app uses to consume the service. Apps and services communicate with one another indirectly via the Cloud Foundry Router.

## Reuse Service, Backing Service

As stated in the beginning of this section, a service is an app including service broker functionality. The service can therefore also run as a microservice component within an application. In this case, the service represents a "Reuse Service". Services, which don't represent components within applications, but rather run "standalone", are referred to as "Backing Services"

## OAuth 2.0, Resource Owner, Client, Resource Server and the Authorization Server (XSUAA Service)

The security functionality of SAP Cloud Platform 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 his 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 Runtime Platform uses a standard implementation of OAuth 2.0 to protect its platform resources (Orgs, Spaces, and platform operations on those entities).

The OAuth 2.0 specification is however reused for SAP Cloud Platform with a proprietary implementation to protect the resources of business web applications powered by the Cloud Foundry Runtime Platform. 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 authorizations. A set or sub-set of these authorizations is assigned to the user after he's authenticated (proven his identity) towards the system. The application also acts as the OAuth 2.0 Resource Server, because it contains the resource app(s). All apps of an application operate under the same OAuth 2.0 Client.

The "Extended Services - User Account and Authentication" (XSUAA) service provides functionality for administrating and assigning application authorizations. It acts as the OAuth 2.0 Authorization Server and represents a typical reuse service. The XSUAA service broker creates a service instance for each application. Each app, which wants to enforce authorizations with the "Security Client Library", is then bound to this XSUAA service instance of the corresponding application.

## **Authentication Against Trusted Parties Only**

The apps of the applications are accessed either via the UI of a user agent or the API(s) 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 XSUAA service have a special "trust relationship", which is established through cross over meta-data 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 users identity. This SAML bearer assertion is presented to XSUAA and the service determines the authorizations of that user. API clients receive their credentials directly from, and authenticate directly against XSUAA.

## **Related Information**

[What Is Authorization and Trust Management? \[page 1096\]](#)

[SAP Cloud Platform Authorization and Trust Management Service in the Cloud Foundry Environment \[page 1088\]](#)

## 6.1.3 Terms and Definitions

This chapter explains important terms for understanding authorization and trust management in SAP Cloud Platform.

Term	Definition
API Access	The use case where a technical user requests the resource of an application with the help of the application programming interface (API) of the resource.
App	An app is a resource of an application, implemented as microservice and represents a deployment unit.
Application	An application is a business web application powered by SAP Cloud Platform.
Application Router	A gateway implementation which can be deployed together with, and used as reverse-proxy for, the application. Application Router provides additional security features (XSRF protection, same origin policy) out of the box.
Application Security Descriptor	A file named xs-security.json which contains the security model an application developer has declared to protect sensitive resources from unauthorized access.
Authentication	The process of identifying an entity, such as a person or system component, usually as a prerequisite for granting the entity access to the system.
Authorities	Authorities represent authorizations which have been granted to an OAuth 2.0 Client to act on its own behalf.
Authorization	The right to perform a given activity in the system. Authorizations can be subdivided into functional authorizations: The right to perform an operation, and instance-based authorizations: The right to perform operations on specific data.
Backing service	Any service that an application consumes over the network as part of its normal operation. Examples include datastores, messaging or queuing systems, and caching systems.
Client Library	Client Libraries provide application programming interfaces (APIs) which applications can utilize to enforce their authorization model.
Identity	One or a combination of several attributes which uniquely identify a user and can be used for authentication.
Multi Tenancy	A multi tenant service/application serves requests from different customers - the tenants - and processes their data strictly isolated from one another.
OAuth 2.0	A specification that describes how to delegate user authorizations to a client. The client acts on the behalf of the user, without the need to reveal user credentials, such as passwords, to the client. OAuth is used by many social network providers and by corporate networks. SAP Cloud Platform has reused the concept with a proprietary implementation to secure application resources.
OAuth 2.0 Authorization Server	The system which assigns scopes to OAuth 2.0 Clients for access to resources belonging to the OAuth 2.0 Resource Owner. In SAP Cloud Platform, the scopes are assigned to the OAuth 2.0 Resource Owner after the user has successfully completed the authentication process.
OAuth 2.0 Client	An entity which holds the scopes to access resources provided by the OAuth 2.0 Resource Server. In SAP Cloud Platform, the role of the OAuth 2.0 Client deviates from the standard, because it is assigned to the application, which also represents the OAuth 2.0 Resource Server.

Term	Definition
OAuth 2.0 Resource Owner	An entity which holds the authorizations to access resources provided by the OAuth 2.0 Resource Server. In SAP Cloud Platform, the role of the OAuth 2.0 Resource Owner deviates from the standard, because the user does not grant his authorizations as scopes towards the OAuth 2.0 Client (the application), but rather receives his application authorizations indirectly from the application after authentication.
OAuth 2.0 Resource Server	The system serving the protected resources. In SAP Cloud Platform, the role of the OAuth 2.0 Resource Server is assigned to the application, because it holds the resources (apps) it is supposed to serve.
OpenID	A decentralized authentication system for web-based services and applications which enables access to all supporting web sites with only one single login, according to the Single Sign-on principle (SSO).
OpenID Connect	An additional layer placed on the OAuth 2.0 protocol stack to enable clients verify the identity of a user with the help of an OAuth 2.0 authorization server.
Platform service	A software that enables, facilitates, or accelerates the development of applications and other platform services on SAP Cloud Platform. 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.
Reuse Service	A service that provides important reuse functionality, which is called from numerous other apps of applications.
Role	The role of a user within the business organization. It contains the values of those authorizations (authorization values) the respective role is eligible to hold.
Role Collection	A role collection contains at least one role and represents the association of roles - which contain the authorization values - to a user or to a user group.
Role Template	A description of a role which a user could represent within the business organization. It combines authorizations matching the business semantics that the role represents.
Runtime Container	The environment to and in which a business application is deployed and operated.
Scopes	Scopes represent authorizations which have been granted to an OAuth 2.0 Client, by an OAuth 2.0 Resource Owner to act on his behalf.
Service	A service is a special type of app which provides reuse functionality for application resources. A service implements a service broker which is responsible for creating instances of the service and binding these to the apps of the consuming application.
Single Sign-on	The principle of a user authenticating himself against different independent applications and systems with one single login and credentials entry.
Token Exchange	A scenario where the calling application obtains a server specific token for the called service in exchange for the application's own token. The server specific token is included in the request towards the called service.
Token Forwarding	A scenario where the calling application includes its own token in the request towards the called service.
Trust	Represents the configuration state between two entities (systems) in which one entity is configured with information of the other (callback endpoints, certificates, etc.) and vice versa.
UAA	The component User Account and Authentication (UAA) - provided by Cloud Foundry - is responsible for managing authorizations related to a Cloud Foundry landscape (Orgs, Spaces, landscape operations, etc.).

Term	Definition
Web Access	The use case where a human user requests the resource of an application with the help of a user interface (UI) or user agent (browser).
XSUAA	The component eXtended UAA (XSUAA) is part of the Authorization and Trust Management service and is responsible for managing authorizations related to the applications running in a Cloud Foundry landscape (application endpoints, application data, etc.).

## 6.1.4 Monitoring and Troubleshooting

This section provides information on troubleshooting-related activities for Authorization and Trust Management 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.

Our troubleshooting information can be found in our Guided Answers [Troubleshooting for Authorization and Trust Management in the Cloud Foundry Environment](#). Guided Answers are a specialized format to guide you step by step through troubleshooting topics. 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](#)
- [Cannot Add Role Templates to Predefined Role Collections](#)

## Related Information

[Getting Support \[page 1161\]](#)

[Enable and Provide Application Logs \[page 1131\]](#)

## 6.1.4.1 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](#), 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 \[page 932\]](#).
3. Choose your organization.

#### i Note

The Cloud Foundry command line interface prompts you to choose an org. To find the org of your subaccount, use the cockpit to go to your subaccount. You find the org in the Cloud Foundry tile under *Organization* (see [Navigate to Orgs and Spaces \[page 917\]](#)).

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 restart <application>
```

 Example

```
cf restart 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 restart <application>
```

 Example

```
cf restart your-app
```

11. Running on the cloud management tools feature set B: Create an incident for your local supportRunning on the cloud management tools feature set A: Create an incident using the component BC-CP-CF-SEC-IAM. Use the [SAP Support Portal](#). For more information, see [Monitoring and Troubleshooting \[page 1130\]](#).
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
<b>SAP_EXT_TRC</b>	Enablement of logs
<b>SAP_EXT_TRL</b>	<p>Log level ranging from off to high</p> <p>Values (integer):</p> <ul style="list-style-type: none"> <li>○ <b>0</b> (off)</li> <li>○ <b>1</b> (default)</li> <li>○ <b>2</b> (medium)</li> <li>○ <b>3</b> (high)</li> </ul>

#### Node.js Environment Variables

Environment Variables	Description
<b>DEBUG</b>	<p>Detailed logs of the security API for Node.js</p> <p>Value (string): <b>xssec*</b></p>

## 6.2 Default Role Collections of SAP Cloud Platform Cloud Foundry Environment [Feature Set B]

The following table displays the default role collections available with cloud management tools feature set B after initially deploying your accounts. For more information, see the related links.

#### 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>	Includes read-write authorizations for updating the global account, setting entitlements, and creating, updating, and deleting subaccounts.
Global Account Administrator	Global Account Usage Reporting Viewer	GlobalAccount_Usage_Reportng_Viewer	uas!<suffix>	Includes read-only authorizations for viewing global account usage information.

Role Collection	Role Name	Role Template	App ID	Role Description
Global Account Administrator	User and Role Administrator	xsuaa_admin	xsuaa!<suffix>	Includes read-write authorizations for trusted identity providers, role collections, roles and users.
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 cloud cockpit.
Subaccount Administrator	Subaccount Admin	Subaccount_Admin	cis-local!<suffix>	Includes read-write authorizations for viewing subaccount entitlements and for creating and deleting environment instances.
Subaccount Administrator	User and Role Administrator	xsuaa_admin	xsuaa!<suffix>	Includes read-write authorizations for trusted identity providers, role collections, roles and users.
Subaccount Administrator	Subaccount Service Administrator	Subaccount_Service_Administrator	service-manager!<suffix>	Administrative access to service brokers and environments on a subaccount level.
Global Account Viewer	Global Account Viewer	GlobalAccount_Viewer	cis-central!<suffix>	Includes read authorizations for viewing subaccount entitlements and for creating and deleting environment instances.
Global Account Viewer	Global Account Usage Reporting Viewer	GlobalAccount_Usage_Reportin g_Vi ewer	uas!<suffix>	Includes read-only authorizations for viewing global account usage information.
Global Account Viewer	User and Role Auditor	xsuaa_auditor	xsuaa!<suffix>	Includes read authorizations for trusted identity providers and users

Role Collection	Role Name	Role Template	App ID	Role Description
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_Viewer	destination-xsapp-name!<suffix>	View destination configurations, certificates and subaccount trust via the Destination editor in the cloud cockpit.
Subaccount Viewer	Subaccount Viewer	Subaccount_Visitor	cis-local!<suffix>	Includes read authorizations for viewing subaccount entitlements and for creating and deleting environment instances.
Subaccount Viewer	User and Role Auditor	xsuaa_auditor	xsuaa!<suffix>	Includes read authorizations for trusted identity providers and users
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 cloud cockpit.
Connectivity and Destination Administrator	Cloud Connector Administrator	Cloud_Connector_Administrator	connectivity!<suffix>	Operate the data transmission tunnels used by the Cloud connector.

Role Collection	Role Name	Role Template	App ID	Role Description
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 cloud cockpit.

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

## 6.3 Principal Propagation

Exchange user ID information between systems or environments in SAP Cloud Platform.

### In This Section

- [Principal Propagation from the Cloud Foundry to the Neo Environment \[page 1143\]](#)
- [Principal Propagation from the Neo to the Cloud Foundry Environment \[page 1136\]](#)

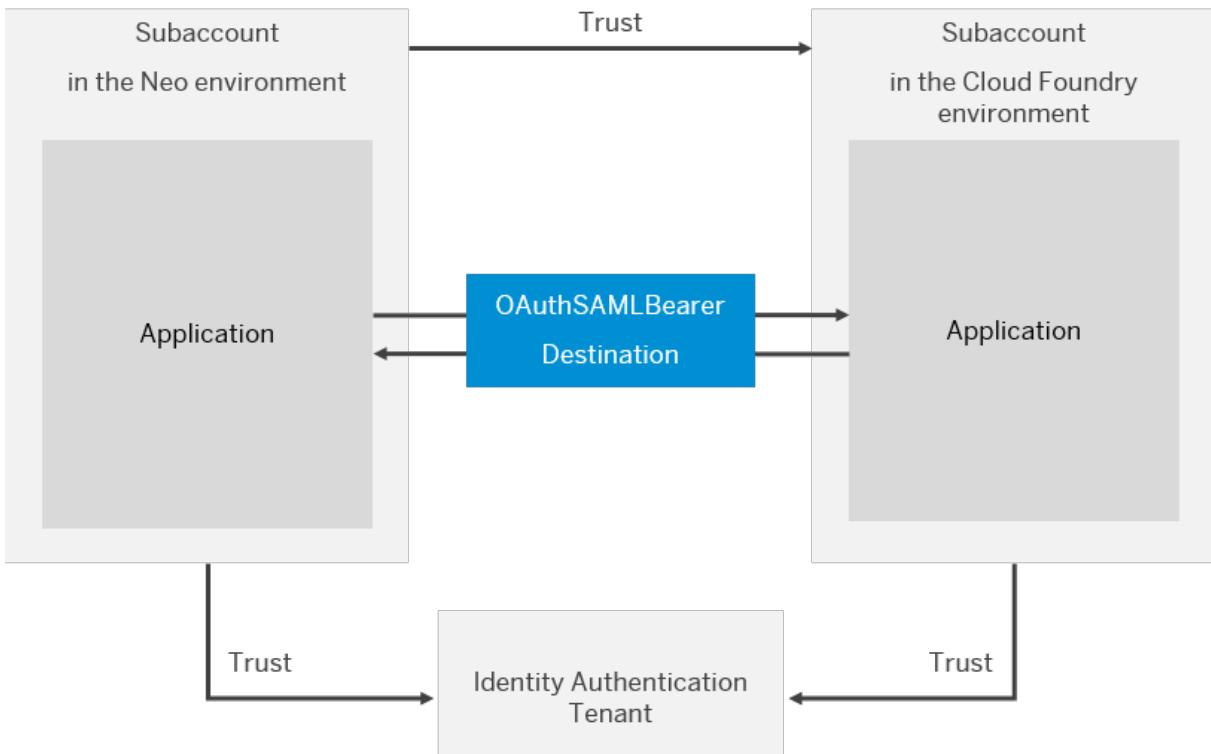
### Other Principal Propagation Scenarios

- [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](#)

### 6.3.1 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 Identity Authentication 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 Cloud Platform 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 SAP Cloud Platform as a Local Service Provider](#).
- Both accounts have a trust configuration to the same Identity Authentication tenant. See:
  - [Identity Authentication Tenant as an Application Identity Provider](#) (for the Neo environment)
  - [Manually Establish Trust and Federation Between UAA and SAP Cloud Platform Identity Authentication Service \[page 794\]](#) (for the Cloud Foundry environment)
- You have developed and deployed both applications, each in the corresponding subaccount.

**i 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 \[page 77\]](#)
- [User Account and Authentication Service of the Cloud Foundry Environment \[page 1101\]](#)
- [Building Roles and Role Collections for Applications \[page 819\]](#)
- [Create a Service Instance from the xsuaa Service \[page 278\]](#)
- [Bind the xsuaa Service Instance to the Application \[page 280\]](#)

#### i 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.

### 6.3.1.1 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.

#### i Note

For the purposes of the example listed in this document, use `pptest` as value.

Field	Description
URL	<p>The URL of the protected resource in the Cloud Foundry environment. See <a href="#">Configuring Application URLs</a>.</p> <p>Example: <code>https://&lt;tenant-specific-route-for-your-business-app&gt;.cfapps.eu10.hana.ondemand.com/</code></p>
Authenticati- on	OAuth2SAMLBearerAssertion
Proxy Type	Internet
Audience	<p>Copy the value of <code>entityID</code> property of the SAML 2.0 metadata representing your subaccount in the Cloud Foundry environment.</p> <div style="background-color: #f0f8ff; padding: 10px;"> <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><code>https://&lt;your subaccount's subdomain&gt;.authentication.&lt;SAP Cloud Platform host&gt;/saml/metadata</code></pre> <p>Example:</p> <pre><code>https://demo.authentication.eu10.hana.ondemand.com/saml/metadata</code></pre> <p>For the &lt;SAP Cloud Platform host&gt;, see <a href="#">Regions and API Endpoints Available for the Cloud Foundry Environment</a>.</p> </div>
	<p>Example of audience/entityID:</p> <pre><code>demo.aws-live-eu10</code></pre>
Client Key	<p>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> ▷.</p>

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: 10px; margin-top: 10px;"> <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;SAP Cloud Platform host&gt;/saml/metadata</pre> <p>Example:</p> <pre>https://demo.authentication.eu10.hana.ondemand.com/saml/metadata</pre> <p>For the &lt;SAP Cloud Platform host&gt;, see <a href="#">Regions and API Endpoints Available for the Cloud Foundry Environment</a>.</p> </div>
Token Service User	<p>Example of token service URL:</p> <pre>https://demo.authentication.eu10.hana.ondemand.com/oauth/token/alias/demo.aws-live-eu10</pre>
Token Service Password	<p>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>.</p>
System User	<p>Empty.</p>

- Save the changes.

### 6.3.1.2 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.

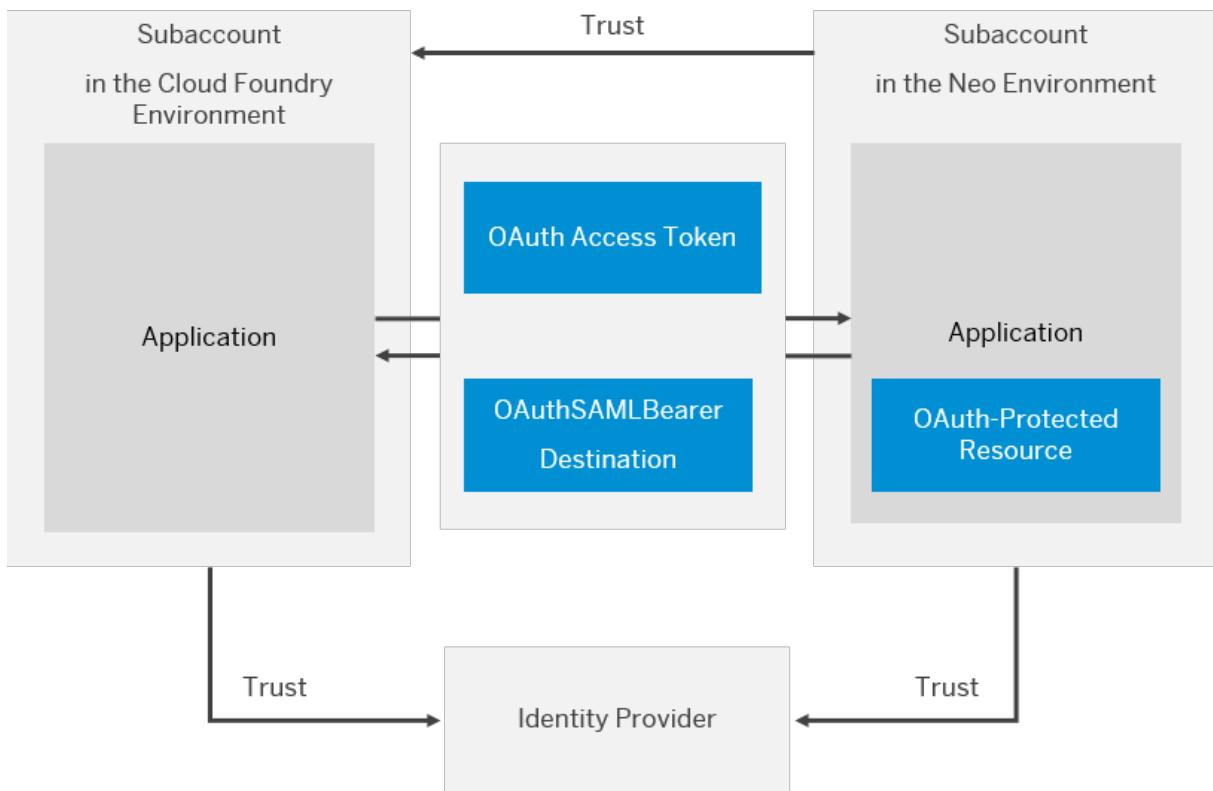
- a. In SAP Cloud Platform cockpit, navigate to the subaccount in the Neo environment. See [Navigate to Global Accounts and Subaccounts \[Feature Set A\] \[page 756\]](#)
  - b. Navigate to .
  - c. Choose [Get Metadata](#) and save the metadata file representing this subaccount.
3. In the subaccount 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 .
    - 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.

### 6.3.2 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 Cloud Platform subaccounts. See [Managing Member Authorizations in the Neo Environment](#).
- You have a *custom* local service provider configuration (this means in **cloud cockpit** **Security** **Trust** **Local Service Provider** you have chosen **Configuration Type** **Custom**) in your subaccount in the Neo environment. See [Configure SAP Cloud Platform as a 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 \[page 798\]](#) (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*. See [Create a Service Instance from the xsuaa Service \[page 278\]](#) and [Bind the xsuaa Service Instance to the Application \[page 280\]](#).
- You have deployed both applications, each in the corresponding subaccount.

### i Note

All configuration steps described in this tutorial are done using the cloud cockpit.

## Contents

- Create Trust Between the Subaccounts [page 1145]
- Create an OAuth Client [page 1146]
- Create a Destination [page 1148]

### 6.3.2.1 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 Cloud Platform cockpit, navigate to the subaccount in the Cloud Foundry environment.
  - b. Navigate to [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 [Trust](#) .
  - c. Choose [Add Trusted Identity Provider](#) and configure the new trust configuration settings:
    - In the [Name](#) field, enter the following:  
<your Cloud Foundry API endpoint host>/<Cloud Foundry subaccount ID>

#### → Tip

You can view the **API endpoint host** and **subaccount ID** from [cloud cockpit](#) > <your global account> > <your subaccount> > <your space> > Overview.

The screenshot shows the SAP Cloud Platform Cockpit interface. On the left, there's a sidebar with navigation links: Overview, Spaces, Subscriptions, Connectivity (which is expanded), Quota Plans, Resource Consumption, and Members. The main content area has a breadcrumb path: Home [Europe (Rot)] / Europe (Frankfurt) / SAP\_connectivity / destination\_validation. Below this, it says "Subaccount: destination\_validation - Overview". Under "Subaccount Details", it shows "Subdomain: destinationvalidation" and "ID: b12...139". To the right, under "Cloud Foundry", it lists "Organization: SAP\_destination\_validation", "Spaces: 1", "Members: 6", and "API Endpoint: https://api.cf.eu10.hana.ondemand.com". There's also a "Disable Cloud Foundry" button at the bottom right of the Cloud Foundry section.

- In the [Signing Certificate](#) field, enter the X509 certificate of the Cloud Foundry account.

**i 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.

### 6.3.2.2 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.

#### Procedure

1. In the cloud cockpit, navigate to the subaccount in the Neo environment.
2. Navigate to   , and choose *Register New Client*.
3. Enter the following information:
  - *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.

## OAuth Settings

Branding   **Clients**   Platform API

### Clients

Register New Client

Client ID	Name	Subscription
937f7df2-99cc-3983-a191-b16e8...	pptest	i032374trial/com

\* Name:  Translations (0)

Description:

\* Subscription:

\* ID:

\* Authorization Grant:

Confidential:

\* Secret:

Skip Consent Screen:

\* Redirect URI:

Token Lifetime:  days Infinite if empty

Refresh Token Lifetime:  days Infinite if empty

#### i Note

When creating the required OAuthSAMLBearer destination later, you will need the following information from the OAuth client you created:

- *ID*
- *Secret*

### 6.3.2.3 Create a Destination

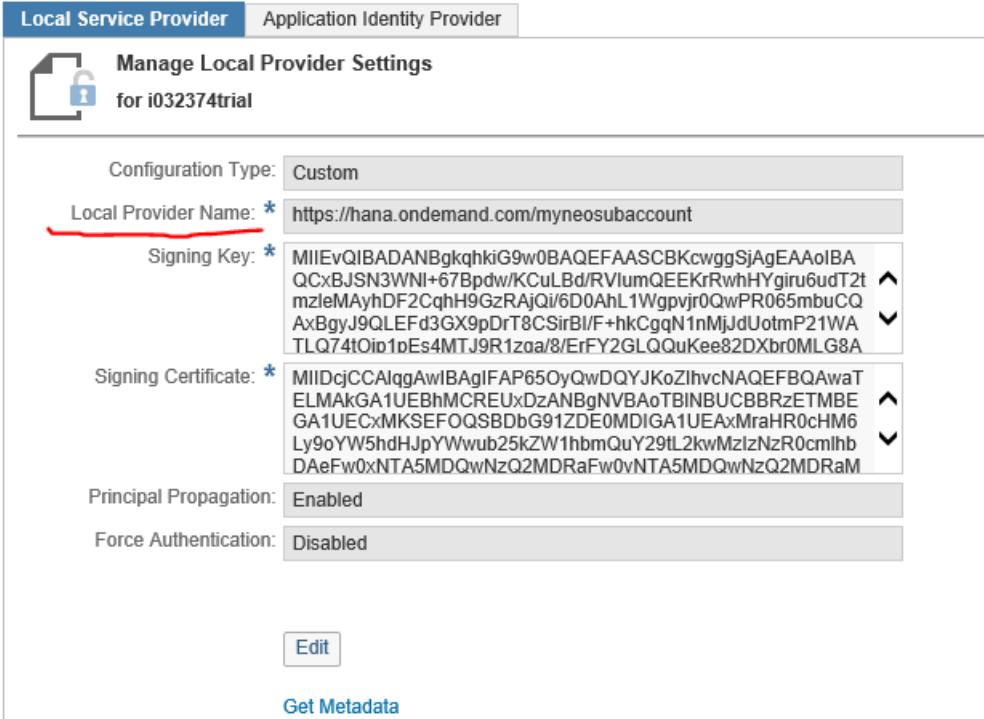
#### Context

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>
Authentica- tion	OAuth2SAMLBearerAssertion
Proxy Type	Internet

Field	Description
Audience	<p>The value of the local service provider name in the subaccount in the Neo environment.</p> <p>Copy the value from  <a href="#">cloud cockpit</a> &gt; &lt;your Neo subaccount&gt; &gt; <a href="#">Security</a> &gt; <a href="#">Trust</a> &gt; <a href="#">Local Service Provider</a> &gt; <a href="#">Local Service Provider Name</a>.</p>  <pre> Local Service Provider Application Identity Provider Manage Local Provider Settings for i032374trial  Configuration Type: Custom Local Provider Name: * https://hana.ondemand.com/myneosubaccount Signing Key: * MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwgjSjAgEAAoIBA QCxBSN3WNI+67Bpdw/KCulBd/RVlumQEEKrRvhHYgiru6udT2t mzleMAyhDF2CqhH9GzRAjQi/6D0AhL1Wgpvjr0QwPR065mbuCQ AxBgyJ9QLEFd3GX9pDrT8CSirBl/F+hkCgqN1nMjjdUotmP21WA TLQ74tOip1pEs4MTJ9R1za/8/ErfY2GLQQuKee82DXbr0MLG8A Signing Certificate: * MIIDcjCCAIqgAwIBAgIFAP65OyQwDQYJKoZIhvcNAQEFBQAwAT ELMAkGA1UEBhMCREUxDzANBgNVBAoTBINBUCBBRzETMBE GA1UECxMKSEFOQSBDbG9IZDE0MDIGA1UEAxMraHR0cHM6 Ly9oYW5hdHJpYWwub25kZW1hbmQuY29tL2kwMzIzNzR0cmhb DAeFw0xNTA5MDQwNzQ2MDRaFw0vNTA5MDQwNzQ2MDRaM Principal Propagation: Enabled Force Authentication: Disabled  Edit Get Metadata </pre>
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: cloud cockpit ► <i>cloud cockpit</i> ► <your Neo subaccount> ► <your application> ► <i>Security</i> ► <i>OAuth</i> ► <i>Branding</i> ▶.						
OAuth Settings	<p><b>Branding</b></p>  <p><b>OAuth URLs</b></p> <table> <tr> <td>Authorization Endpoint:</td> <td><a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize</a></td> </tr> <tr> <td>Token Endpoint:</td> <td><a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token</a></td> </tr> <tr> <td>End User UI:</td> <td><a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2</a></td> </tr> </table>	Authorization Endpoint:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize</a>	Token Endpoint:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token</a>	End User UI:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2</a>
Authorization Endpoint:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/authorize</a>						
Token Endpoint:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2/api/v1/token</a>						
End User UI:	<a href="https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2">https://oauthasservices-myneosubaccount.hana.ondemand.com/oauth2</a>						
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)	authnContextClass <input type="button" value="..."/> jknhjlnh <input type="button" value="Delete"/>
<b>*URL:</b>	https://myneoadapp.hana.ondemand.com/myprotectedresource/	<input checked="" type="checkbox"/> Use default JDK truststore
<b>Proxy Type:</b>	Internet	
<b>Authentication:</b>	OAuth2SAMLBearerAssertion	
<b>*Audience:</b>	https://hana.ondemand.com/myneosubaccount	
<b>*Client Key:</b>	*****	
<b>*Token Service URL:</b>	https://oauth2services-myneosubaccount.hana.ondemand.co	
<b>Token Service User:</b>	937f7df2-99cc-3983-a191-b16e88203c0b	
<b>Token Service Password:</b>	*****	
<b>System User:</b>		

4. Save the changes.

## 6.4 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 is necessary to consider compliance with industry-specific legislation in different countries.

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-specific requirements. Furthermore, this information should not 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.

### i 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.

## SAP Cloud Platform

This documentation covers personal data relating to SAP Cloud Platform accounts and data stored in databases by SAP Cloud Platform. SAP Cloud Platform 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 have 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 \[page 917\]](#).
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 Platform Identity Authentication service.
- Platform users are stored in platform identity provider, a tenant of SAP Cloud Platform Identity Authentication service, or your own identity provider.
- Business users are stored in a tenant of SAP Cloud Platform Identity Authentication service or your own identity provider.

### [Glossary for Data Protection and Privacy \[page 1153\]](#)

The following terms are general to SAP products. Not all terms may be relevant for SAP Cloud Platform.

### [Change Logging and Read-Access Logging \[page 1154\]](#)

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.

### [Personal Data Record \[page 1155\]](#)

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.

### [Deletion \[page 1155\]](#)

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.

### [Consent \[page 1158\]](#)

SAP Cloud Platform supports you in collecting and managing the consent of data subjects in the following ways:

## 6.4.1 Glossary for Data Protection and Privacy

The following terms are general to SAP products. Not all terms may be relevant for SAP Cloud Platform.

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.
Data subject	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 personal data 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 personal data is no longer required for the primary business purpose. After the EoP has been reached, the data is blocked and can only be accessed by users with special authorization, for example, tax auditors.
<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.

Term	Definition
<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, or trade union membership and the processing of genetic data, biometric data, data concerning health, sex life or sexual orientation or personal data concerning bank and credit accounts</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>
<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>• Access control: Authentication features</li> <li>• Authorizations: Authorization concept</li> <li>• Read access logging</li> <li>• Transmission control/communication security</li> <li>• Input control/change logging</li> <li>• Availability control</li> <li>• Separation by purpose: Subject to the organizational model implemented and must be applied as part of the authorization concept.</li> </ul>

## 6.4.2 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 environment, see [Audit Log Retrieval API Usage for the Cloud Foundry Environment \[page 988\]](#).

### i Note

For any applications you develop, you must ensure they include logging functions. SAP Cloud Platform does not provide audit logging functions for custom developments.

Logs are deleted regularly.

### 6.4.3 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 Cloud Platform, 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 \[page 917\]](#).
2. In the navigation area, choose *Members*.

To see the personal data that is used for application logging within SAP Cloud Platform, 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 Platform Identity Authentication service.

For more information, see [Information Report](#) in the SAP Cloud Platform Identity Authentication service documentation.

For SAP Cloud Platform 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, see the [User Management \(SCIM\) API](#) on SAP API Business 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.

### 6.4.4 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 90 days.
- Productive accounts expire based on the terms of your contract.

To deactivate or delete users, see [Erasure](#) in the SAP Cloud Platform Identity Authentication service documentation.

For SAP Cloud Platform Cloud Foundry environment, the User Account and Authentication service creates shadow users to issue tokens for their corresponding users.

For more information about deleting shadow users, see [Delete Shadow Users for Data Protection and Privacy \[page 1157\]](#).

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 Cloud Platform 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 Cloud Platform 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.

#### i Note

If your data is stored outside SAP Cloud Platform, 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.

## Related Information

[Global Account Termination \[page 776\]](#)

## 6.4.4.1 Delete Shadow Users for Data Protection and Privacy

The User Account and Authentication service of the Cloud Foundry environment stores shadow users, which contain personal data. Data privacy regulations or policies may require you to delete this data, for example, when the user has left your organization.

### Prerequisites

The security administrator must have the following scopes:

- `xs_user.read`
- `xs_user.write`

### Context

#### i Note

When handling personal data, consider the legislation in the various countries 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.

Whenever a user authenticates at the default identity provider or at an SAML 2.0 identity provider, the User Account and Authentication service stores user-related data records in the form of shadow users. This condition applies to platform users and business users. The UAA uses the information of the shadow users to issue tokens that refer to the specific user. For more information about shadow users, see the Cloud Foundry documentation.

#### → Tip

As an administrator, you can switch off the creation of shadow users in the trust configuration of your identity provider. However, users can't log in until you explicitly create their shadow users. For more information, see the related link.

To delete shadow users, you set up access to the API and then use the SCIM REST APIs to retrieve and delete shadow users.

Use the following procedure:

### Procedure

1. Enable API access to your subaccount.

For more information, see [Get API Access \[page 912\]](#).

2. To retrieve and delete the shadow users, use the `GET` and `DELETE` methods for the System for Cross-domain Identity Management (SCIM) interface of the `platform` API of the Authorization and Trust management service.

For more information about the `platform` API, see <https://api.sap.com/package/authtrustmgmnt> on SAP API Business Hub.

## Related Information

[Data Protection and Privacy \[page 1151\]](#)

<https://docs.cloudfoundry.org/>

[Switch Off Automatic Creation of Shadow Users \[page 812\]](#)

## 6.4.5 Consent

SAP Cloud Platform supports you in collecting and managing the consent of data subjects in the following ways:

SAP Cloud Platform Identity Authentication service provides tools to manage privacy policies and terms of use agreements.

For more information, see [Configuring Privacy Policies in SAP Cloud Platform Identity Authentication Service](#) and [Configuring Terms of Use](#).

See also [Consent](#) in the SAP Cloud Platform Identity Authentication service documentation.

## 6.5 Security in the Kyma Environment

The Kyma environment-specific security aspects include guidelines on personal data protection and details on processing and storing logs.

## Operational Logs

Operational logs are not relevant for audit. These are typically system-related activities and operations you can analyze to get more information about performance or inspect potential problems.

## 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 case of a breach.
- Configuration changes introduced to the system, applications, and components.

### i Note

Information recorded by audit logs may include personal data, such as user IDs, or confidential data, such as logins or email addresses. For details on how to handle such sensitive data, see [Data Protection and Privacy in the Kyma Environment \[page 1159\]](#).

For general information on data protection and privacy, see [Data Protection and Privacy \[page 1151\]](#).

### 6.5.1 Function Security

When creating Functions, make sure they comply with the security standards and be aware of potential issues.

To eliminate potential security risks when using Functions, bear in mind these few facts:

- By default, JSON Web Tokens (JWTs) issued by Dex 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

### 6.5.2 Data Protection and Privacy in the Kyma Environment

Protect any confidential and personal data from leakage or misuse by not storing or processing it in an unsafe manner.

## Data Storage and Processing

Kyma environment provides functionality that allows you to store and process data, such as configuration files or specifications. To protect the confidentiality of your information, store only public data. Do not use the environment to process and store any kind of confidential information, including personal data.

## Personal Data Use in Functions and Event Payloads

Do not store any personal data in Functions or event payloads they receive, as they are not logged. If the Function calls an external API, receives data from an external API, or processes events, ensure compliance with the data privacy laws by making sure that the data is properly logged.

## Logging Personal Data

The audit log records personal data used by the Kyma environment. On average, it stores the data for at least 201 days, and in case of some services it can be prolonged to up to 400 days. If you want to store any other personal data, be cautious and bear in mind the recommendations provided in [Data Protection and Privacy \[page 1151\]](#).

### i Note

The logging systems used by the Kyma environment collect logs from stdout. Therefore, you should not log your personal data to stdout, but make sure it is logged using a proper audit logging service.

## Retrieving Personal Data and Audit Log Entries

You can retrieve information regarding your personal data stored in the audit log system, provided that is stored there. To do so, open a [support ticket](#).

# 7 Getting Support

Use SAP Community, get guided answers, or explore SAP Support Portal.

Depending on your global account, you can use the following support media:

Trial Accounts	Customer and Partner Accounts
<ul style="list-style-type: none"><li>• <a href="#">SAP Cloud Platform Community</a></li><li>• <a href="#">Guided Answers for SAP Cloud Platform</a></li></ul>	<ul style="list-style-type: none"><li>• <a href="#">SAP Cloud Platform Community</a></li><li>• <a href="#">Guided Answers for SAP Cloud Platform</a></li><li>• <a href="#">SAP Support Portal</a> (An S-user is required to log in and file an incident.)</li></ul>

To report an incident (issue) in the SAP Support Portal, proceed as follows.

## 1. Check Platform Status

### Context

#### [AWS, Azure, or GCP Regions]

Check the availability of the platform at [SAP Trust Center](#). You can check the following:

- the availability by service on the [SAP Cloud Platform](#) tile of the [Cloud Status](#) tab page;
- the availability by region on the [Data Center](#) tab page.

For more information about selected platform incidents, see [Root Cause Analyses](#).

#### [China (Shanghai) Region]

Check the availability of the platform at <https://status.cn40.platform.sapcloud.cn/>.

#### [Government Cloud (US) Region]

Planned downtimes and outage communication are sent through e-mail to the initial administrator of your global account.

## 2. Check Tools Versions

### Context

Make sure that you have 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](#).

## 3. Log on to SAP Support Portal

### Procedure

1. Open [SAP Support Portal](#).
2. Choose *Report an Incident*.  
The SAP ONE Support Launchpad opens.
3. Perform a search to check whether a similar incident has already been reported.
4. If you still need to report an incident, choose [Contact SAP Support](#).
5. Enter the customer number related to your SAP Cloud Platform contract.
6. Enter your S-user (example: **s1234567890**). A form opens where you fill in details about the incident.

## 4. Identify Affected System

### Context

Specify the affected system. For more information, see KBA [2848890](#).

#### i Note

When you specify the correct system/product, the correct support SLA is applied to your case.

Not choosing the appropriate system/product may negatively affect the processing of the incident.

## 5. Provide Incident Details

### Procedure

1. Select the language, set priority of the incident, and enter a subject. If you set a high or very high priority, you must also describe the business impact of the incident.
2. To help support staff process your issue as fast as possible, fill in the *Description* field:
  - For the Cloud Foundry environment, provide:
    - Region and global account name. In the cockpit, open the affected subaccount, and copy the URL.
    - Java application name and URL (if the problem is related to Java applications). In the cockpit, open the respective Java application's Overview page.
    - Database-related details based on your environment and infrastructure provider (if the problem is related to SAP HANA). See [Providing Details for Database Problems in the Cloud Foundry Environment](#) [page 1163].

- For the Kyma environment, provide:
    - the Subaccount name, or
    - the URL to the Kyma Console
3. Select the component name of the area that best fits your issue. Selecting the right component directs the issue to the corresponding support team. To check the complete list of components, see [Support Components \(prefiltered for the Cloud Foundry environment\)](#).

**i Note**

For the Kyma environment issues, select **BC-CP-XF-KYMA**.

4. Enter the steps to reproduce the issue and if necessary, add some attachments.
5. (Optional) Define any additional contacts, apart from the reporter (who is filled in automatically).
6. Choose **Submit** to create the incident.

**i Note**

If you have problems creating and sending an incident, or your ticket isn't processed as fast as you need, contact the 24/7 phone hotlines. See SAP Note [560499](#).

## Related Information

[Cloud Management Tools — Feature Set Overview](#)

[Gather Support Information \[page 1165\]](#)

[Platform Updates and Notifications in the Cloud Foundry Environment \[page 1165\]](#)

[Root Cause Analyses](#)

## 7.1 Providing Details for Database Problems in the Cloud Foundry Environment

If your problem is related to a 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 cockpit, navigate to the org and space the database is provisioned in.</li> <li>2. In the navigation area, go to  <a href="#">SAP HANA</a>  <a href="#">Database Systems</a></li> </ol>

Infrastructure Provider	Details You Need to Provide	How to Find the Details You Need
AWS regions  (databases provisioned before June 4, 2018)		<ol style="list-style-type: none"> <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 cockpit, navigate to the space the SAP HANA service instance is provisioned in.</li> <li>2. In the navigation area, choose  <b>Services</b> <b>Service Instances</b>.</li> <li>3. In the list of available services, find your SAP HANA service instance and choose  <b>Open Dashboard</b> from the <b>Actions</b> column.</li> <li>4. Copy the service instance GUID under  <b>Details</b> <b>ID</b>. Example GUID:  <code>ad539b83-39bd-4ff4-8b41-1a5dfdfca7ea</code> </li> </ol>
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](#)

## 7.2 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 ZIP file to it, and send it to the relevant OSS component. For more information, see [Getting Support \[page 1161\]](#).

## 7.3 Platform Updates and Notifications in the Cloud Foundry Environment

SAP Cloud Platform is a dynamic product, which has continuous production releases (updates). To get notifications for the new features and fixes every release, subscribe at the [SAP Community wiki](#) by logging on and choosing the [Watch](#) icon.

## Reasons for Region Updates

SAP Cloud Platform regions are updated in the following cases:

- **Biweekly updates** (standard) - aligned with the contractual obligations of SAP Cloud Platform to customers and partners. Such updates usually don't affect productive applications, because most SAP Cloud Platform services support zero downtime maintenance. For more information about the biweekly updates of cloud management tools feature set A, see [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 of cloud management tools feature set A, see [Service Level Agreement for SAP Cloud Services](#). We let you know about these upgrades one week 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 [SAP Cloud Platform](#) tile of the [Cloud Status](#) tab page;
- the availability by region on the [Data Center](#) tab page.

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](#).

## 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](#)

[Cloud Management Tools — Feature Set Overview](#)

## 7.4 Operating Model in the Cloud Foundry Environment

An operating model clearly defines the separation of tasks between the operator and the customer during all phases of an integration project.

SAP Cloud Platform and its services have been developed on the assumption that specific processes and tasks are the responsibility of the customer. The following table contains all processes and tasks involved in operating the platform and the services and specifies how the responsibilities are divided between the operator 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* (release notes) 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 the operator and the customer for the purchase of any of the services in scope. In the event of a conflict, the contractual agreement between the operator 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.

The responsibilities for operating SAP Cloud Platform are listed in the following service catalog.

### Service Catalog

Process	Task	Operator	Customer
Communication Management	Appoint an English-speaking contact person and communicate the name to the operator.  This is required to ensure timely processing of configuration change requests affecting the customer system, interacting with the operator for efficient incident processing, and other interaction between the operator and the customer.		x
	Subscribe to the communication channels offered by the operator for receiving prompt information about any service disruptions, critical maintenance activities affecting the customer system, and change requests requiring action on the customer side.		x

Process	Task	Operator	Customer
	Inform the customer about service disruptions and critical maintenance activities affecting the customer system.	x	
Asset Management	Management of the hardware and infrastructure resources in the region, from acquisition through disposal. This includes the request and approval process, procurement management, life-cycle management, and disposal management.	x	
	Protect IT assets such as systems, network, and data from threats that arise from unauthorized physical access or physical influence on those assets.	x	
Provisioning	Provisioning of 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 the operator's application software and the proper initial configuration of quotas, service subscriptions, permissions, and trust configuration.	x	
	Provide quota according to the ordered package and subscriptions that can be used to enable resources and services (for example, subscribing to an SAP Cloud Platform service).		x

Process	Task	Operator	Customer
Change Management	<p>Apply regular product increments, as well as corrections to the infrastructure, systems, and services to avoid incidents with minimal possible disruption of normal operations. Ensure that all platform changes (such as updates of the Java 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>	x	
	<p>Perform updates of the infrastructure, systems, and services in a biweekly cycle if required. Respectively, for selected services (such as SAP HANA and SAP ASE), offer self-services for applying controlled updates of new versions. Emergency changes, for example, triggered by Incident Management processes, have accelerated testing, approval, and implementation.</p>	x	

Process	Task	Operator	Customer
	<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 (Java application runtimes or databases), in case the patches don't require downtime, or if the vulnerable system puts at risk the operator or other customers.</li> <li>• Inform the customers about the availability of security patches.</li> </ul>	x	
	Adopt the latest patches or updates via the available self-services and by restarting applications when necessary. For example, when a security issue arises.		x
Incident Management	Process incidents reported by the customer according to the Service Level Agreement. The incident is recorded and prioritized in the incident tracking system. Monitor the status and progress of the incident throughout its whole lifecycle and give regular status updates to the customer.	x	

Process	Task	Operator	Customer
	In the event of incidents, make reasonable effort to support end users and manage their incidents, to explore self-help tools to find already documented solutions, and to liaise with operator support in the event of new problems to ensure timely processing of incidents affecting the resources in the customer account.		x
	Confirm incident resolution in the incident tracking system.		x
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.	x	
	Confirm service request completion in the service request tracking system.		x
Backup & Restore	Perform a backup of the database systems hosted in the subaccount. A database log backup is done every 10 minutes and stored on the primary storage. Every 2 hours the logs are transferred from primary to secondary storage. Full data backup is done every day.	x	

Process	Task	Operator	Customer
	Restore previously backed-up data to recover to a consistent state. Verify the completeness of the restored data based on log files created during the recovery and smoke tests to verify the system's consistency.	x	
	Give regular status updates to the customer throughout the entire restore procedure.	x	
	Collaborate with the operator to ensure timely processing of data restores if required.		x
	Validate logical integrity and consistency of the restored data.		x
User Access Management	Manage users, permissions, and security configurations within the subaccount.		x
System Monitoring	Ensure availability of the customer system according to the Service Level Agreements as agreed in the contractual agreement between the operator and the customer, by active monitoring, prompt issue detection, and incident prevention.	x	
	Monitor the resource consumption (memory, CPU, storage) to detect issues in technical operations.	x	
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 the operator.	x	

Process	Task	Operator	Customer
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.		x
	Provide infrastructure, tools, and application programming interfaces for the lifecycle management and operations of the application in the subaccount.	x	
	Regularly adopt the latest versions of the tools for lifecycle management and operations offered at the <a href="#">SAP Development Tools site</a> .		x
Network Management	Manage the network isolation of the subaccounts provisioned to the customer.	x	
	Operate the network infrastructure transparently for customers, ensuring elasticity, high availability, and security.	x	
	Create and manage own Web domain for the application in the subaccount to ensure data isolation.		x

Process	Task	Operator	Customer
Penetration Testing	Inform the operator 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 the operator and the customer aren't to be shared with any person or entity without explicit written authorization from the operator. Customers are required to share the results with the operator and work together with the operator's operations to mitigate or remedy any security issues.		x
Decommissioning	Ensure the secure deletion of data and hardware disposal. This includes the disassembly 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.	x	

## 7.5 Support Components

A list of support components for SAP Cloud Platform 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.

### i Note

The table below lists the support components for SAP Cloud Platform services. If you are looking for components of tools or issues related to software logistics, see [Additional Components \[page 1283\]](#).

## Service Components

Service Availability

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Support	Release Status
							Available as Trial	
Agentry	Develop and run Agentry metadata-driven mobile applications.	MOB-CLD-AGS	Neo	Extension Suite - Digital Experience	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) Russia (Moscow) UAE (Dubai)		Available

Service Name	Short Description	Support			Available as Trial	Release Status
		Component	Environment	Capability		
Alert Notification	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow)

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
<a href="#">Alert Notification</a>	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available
						US East (VA)		
						Japan (Tokyo)		
						Singapore		
						Australia (Sydney)		
						Canada (Montreal)		
						Brazil (São Paulo)		
<a href="#">Alert Notification</a>	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available
						US West (WA)		
						US East (VA)		
						Singapore		
						Japan (Tokyo)		
<a href="#">Alert Notification</a>	Create and receive real-time alerts about your services	BC-CP-LCM-ANS	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Neo	Integration Suite	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Brazil (São Paulo) Russia (Moscow) Canada (Toronto) UAE (Dubai) KSA (Riyadh )	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
API Management	Expose your data and processes as APIs and manage their lifecycles.	OPU-API-OD	Cloud Foundry	Integration Suite	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	Integration Suite	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	Integration Suite	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Mobile App and Device Management	Manage your mobile devices.	MOB-SEC	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) US East (Sterling)	Yes	Available
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AU-TO-SCALE	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AU-TO-SCALE	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AU-TO-SCALE	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
Application Autoscaler	Automatically increase or decrease the number of application instances.	BC-CP-CF-AU-TO-SCALE	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Application Logging	Create, store, access, and analyze application logs.	BC-NEO-LOG	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh )	Yes	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Application Logging	Create, store, access, and analyze application logs.	BC-CP-CF-AP-PLOG	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available
Application Logging	Create, store, access, and analyze application logs.	BC-CP-CF-AP-PLOG	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
Application Logging	Create, store, access, and analyze application logs.	BC-CP-CF-AP-PLOG	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available
Application Logging	Create, store, access, and analyze application logs.	BC-CP-CF-AP-PLOG	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Authorization and Trust Management	Manage application authorizations and trusted connections to identity providers.	BC-NEO-SEC-IAM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Authorization and Trust Management	Manage application authorizations and trusted connections to identity providers.	BC-XS-SEC	Cloud Foundry	Extension Suite - Development Efficiency	AWS	US East (VA)	Yes	Available
						Europe (Frankfurt)		
						Brazil (São Paulo)		
						Japan (Tokyo)		
						Australia (Sydney)		
						Singapore		
						Canada (Montreal)		
Authorization and Trust Management	Manage application authorizations and trusted connections to identity providers.	BC-XS-SEC	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available
						US West (WA)		
						US East (VA)		
						Singapore		
						Japan (Tokyo)		
Authorization and Trust Management	Manage application authorizations and trusted connections to identity providers.	BC-XS-SEC	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component	Environment						
Authorization and Trust Management	Manage application authorizations and trusted connections to identity providers.	BC-XS-SEC	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available	
Big Data Services	Ready-to-use, fully managed Hadoop and Spark.	BC-NEO-BDS		Extension Suite - Development Efficiency	SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available	
Blockchain Application Enablement	Deliver blockchain-based services on any connected blockchain network.	BC-BCS-VAS	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
SAP Build	Create interactive prototypes based on end-user feedback without code writing.	MOB-UIA-BLD-ADM	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) UAE (Dubai) Australia (Sydney) Japan (Tokyo) Russia (Moscow) Canada (Toronto) Brazil (São Paulo) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Business Entity Recognition	Detect and highlight entities from unstructured text using machine learning.	CA-ML-BER	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Neo	Extension Suite - Digital Process Automation	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) Russia (Moscow) 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)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Singapore Japan (Tokyo) Brazil (São Paulo)	Yes	Available	
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	Extension Suite - Digital Process Automation	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available	
Business Rules	Enrich cloud offerings with decision modeling, management, and execution service.	LOD-BPM-RUL	Cloud Foundry	Extension Suite - Digital Process Automation	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Process Integration	Seamlessly orchestrate business processes and integrate data in real time.	LOD-HCI-PI-OPS-PROV	Neo	Integration Suite	SAP	Europe (Rot)*		Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Process Integration	Seamlessly orchestrate business processes and integrate data in real time.	LOD-HCI-PI-OPS-PROV	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt)	Yes	Available
						US East (VA)		
						Australia (Sydney)		
						Singapore		
						Japan (Tokyo)		
						Canada (Montreal)		
						Brazil (São Paulo)		
Process Integration	Seamlessly orchestrate business processes and integrate data in real time.	LOD-HCI-PI-OPS-PROV	Cloud Foundry	Integration Suite	Azure	Europe (Netherlands)	Yes	Available
						US West (WA)		
						US East (VA)		
						Singapore		
						Japan (Tokyo)		
Process Integration	Seamlessly orchestrate business processes and integrate data in real time.	LOD-HCI-PI-OPS-PROV	Cloud Foundry	Integration Suite	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Connectivity	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Neo	Integration Suite	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Connectivity	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available
Connectivity	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Cloud Foundry	Integration Suite	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
Connectivity	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Cloud Foundry	Integration Suite	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support						
		Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
Connectivity	Establish connections between cloud applications and on-premise systems.	BC-NEO-CON	Cloud Foundry	Integration Suite	Alibaba	China (Shanghai)**	Yes	Available
Continuous Integration and Delivery	Configure and run predefined pipelines for continuous integration and delivery.	BC-CP-CF-CICD	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Data Integration	Integrate data between on-premise and cloud applications on a scheduled (batch) basis.	LOD-HCI-DS	Neo	Integration Suite	SAP	UAE (Dubai) Australia (Sydney) Europe (Rot)* Japan (Tokyo) Russia (Moscow) KSA (Riyadh ) US West (Colorado Springs )	Available	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Credential Store	Store and retrieve credentials such as cryptographic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	Extension Suite - Development Efficiency	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 cryptographic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Credential Store	Store and retrieve credentials such as cryptographic keys and passwords.	BC-CP-CF-SEC-CPG	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support			Available as Trial	Release Status
		Component	Environment	Capability		
Custom Domain	Configure and expose your application under your own domain.	BC-NEO-INFRA	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Custom Domain	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available
Custom Domain	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Custom Domain	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support			Available as Trial	Release Status	
		Component	Environment	Capability			
Custom Domain	Configure and expose your application under your own domain.	BC-CP-CF-SEC-DOM	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes Available
Data Attribute Recommendation	Apply machine learning to match and classify data records automatically.	CA-ML-MEH	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes Available
Data Enrichment	Create or enrich master data using trusted third-party data.	LOD-MDM-DE	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes Available
Data Quality Services	Embed data quality services to validate addresses and enrich with geocodes.	EIM-DQM-SVS	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt)	Yes Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Java Debugging	Debug your Java application even through networks with high latency.	BC-JVM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Destination	Retrieve information about destinations in the Cloud Foundry environment.	BC-NEO-CON	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal) US East (VA)	Yes	Available	
Destination	Retrieve information about destinations in the Cloud Foundry environment.	BC-NEO-CON	Cloud Foundry	Integration Suite	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available	
Destination	Retrieve information about destinations in the Cloud Foundry environment.	BC-NEO-CON	Cloud Foundry	Integration Suite	GCP	US Central (IA)	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Destination	Retrieve information about destinations in the Cloud Foundry environment.	BC-NEO-CON	Cloud Foundry	Integration Suite	Alibaba	China (Shanghai)**	Yes		Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Mobile Services, users	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Neo	Extension Suite - Digital Experience	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) Russia (Moscow)	Yes	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
<a href="#">Mobile Services, users</a>	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	AWS	Australia (Sydney)	Yes	Available
						Brazil (São Paulo)		
						Japan (Tokyo)		
						US East (VA)		
						Europe (Frankfurt)		
						Singapore		
<a href="#">Mobile Services, users</a>	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands)	Yes	Available
						US West (WA)		
						Japan (Tokyo)		
						US East (VA)		
						Singapore		
<a href="#">Mobile Services, users</a>	Build and run mobile apps for B2E and B2B use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Neo	Extension Suite - Digital Experience	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) Russia (Moscow)		Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	AWS	Australia (Sydney) Brazil (São Paulo) Japan (Tokyo) US East (VA) Europe (Frankfurt) Singapore	Yes	Available
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo) US East (VA) Singapore	Yes	Available
Mobile Services, consumers	Build and run mobile apps for B2C use cases.	MOB-CLD-OPS	Cloud Foundry	Extension Suite - Digital Experience	Alibaba	China (Shanghai)**	Yes	Available
Document Classification	Classify business documents automatically using machine learning.	CA-ML-BDP	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Document Information Extraction	Use machine learning to automate your document information extraction processes	CA-ML-DOX	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) Japan (Tokyo) US East (VA)	Yes	Available	
Document Management,integration option	Provide API and UI based document management capabilities to your business applications.	BC-CP-CF-SDM	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal)	Yes	Available	
Document Management,integration option	Provide API and UI based document management capabilities to your business applications.	BC-CP-CF-SDM	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore	Yes	Available	

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Document Service	Store and manage your documents.	BC-NEO-ECM-DS	Neo	Extension Suite - Digital Experience	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Agent Activation for Dynatrace	Connect your Java applications to a Dynatrace SaaS monitoring environment.	BC-NEO-MON-APM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow)	Yes	Available

Service Name	Short Description	Support				Region	Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure			
Enhanced Disaster Recovery	Restore your cloud production environment with minimal data loss.	BC-NEO-DR	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney)		Available
Enterprise Messaging	Connect applications, services and systems across different landscapes.	BC-CP-CF-MES	Cloud Foundry	Integration Suite	AWS	US East (VA) Australia (Sydney) Singapore Brazil (São Paulo) Japan (Tokyo) Europe (Frankfurt) Canada (Montreal)	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Enterprise Messaging	Connect applications, services and systems across different landscapes.	BC-CP-CF-MES	Cloud Foundry	Integration Suite	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available	
Enterprise Messaging	Connect applications, services and systems across different landscapes.	BC-CP-CF-MES	Cloud Foundry	Integration Suite	Alibaba	China (Shanghai)**	Yes	Available	
Extension Factory, serverless runtime	Allows you to create, manage, configure extensions on SAP Cloud Platform	BC-CP-XF-SRT	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal)	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Extension Factory, serverless runtime	Allows you to create, manage, configure extensions on SAP Cloud Platform	BC-CP-XF-SRT	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
Feature Flags	Control the rollout of new features.	BC-CP-CF-FEA-TUR-EFLG	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available	

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Feature Flags	Control the rollout of new features.	BC-CP-CF-FEA-TUR-EFLG	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available	
Feature Flags	Control the rollout of new features.	BC-CP-CF-FEA-TUR-EFLG	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
Feature Flags	Control the rollout of new features.	BC-CP-CF-FEA-TUR-EFLG	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component	Environment						
SAP Fiori Cloud	Revamp your user experience with SAP Fiori on SAP Cloud Platform.	EP-CPP-NEO-OPS	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)*	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
SAP Fiori Mobile	Optimize, build, manage, and monitor SAP Fiori apps on mobile devices.	MOB-FM	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) US East (Sterling) Canada (Toronto) Russia (Moscow) Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
SAP Forms by Adobe	Generate print and interactive forms using Adobe Document Services.	BC-SRV-FP-CLD	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Russia (Moscow) Brazil (São Paulo) Canada (Toronto) UAE (Dubai) Europe (Amsterdam) KSA (Riyadh ) Europe (Frankfurt) US West (Colorado Springs )	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Gamification	Introduce gamification concepts into your applications.	BC-NEO-SVC-GAM	Neo	Extension Suite - Digital Experience	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Git Service	Store and version source code in Git repositories.	BC-NEO-GIT	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
<a href="#">HTML5 Applications</a>	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) ) Australia (Sydney) Japan (Tokyo) Russia (Moscow) Brazil (São Paulo) Canada (Toronto) Europe (Amsterdam) UAE (Dubai)	Yes	Available

Service Name	Short Description	Component	Support		Infrastructure	Region	Available as Trial	Release Status
			Environment	Capability				
<a href="#">HTML5 Applications</a>	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal) Japan (Tokyo)	Yes	Available
<a href="#">HTML5 Applications</a>	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
<a href="#">HTML5 Applications</a>	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available
<a href="#">HTML5 Applications</a>	Develop and run HTML5 applications in a cloud environment.	BC-CP-CF-HTML5	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support			Region	Available as Trial	Release Status	
		Component	Environment	Capability				
Hyperledger Fabric	Create Hyperledger Fabric nodes and connect them to a blockchain network.	BC-BCS-HL	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Identity Authentication - Additional Tenant	Secure authentication and single sign-on for users in the cloud.	BC-IAM-IDS		Extension Suite - Development Efficiency	SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		Available
Identity Authentication	Secure authentication and single sign-on for users in the cloud.	BC-IAM-IDS		Extension Suite - Development Efficiency	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	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Identity Provisioning	Manage identity life-cycle processes for cloud and on-premise systems.	BC-IAM-IPS	Neo	Extension Suite - Development Efficiency	SAP	UAE (Dubai) Australia (Sydney) Brazil (São Paulo) Canada (Toronto) Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) Japan (Tokyo) Russia (Moscow) KSA (Riyadh ) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs )	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Intelligent Product Design	Accelerate product innovation with instant collaboration and live product intelligence.	PLM-DC	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt)		Available	
Intelligent Product Design	Accelerate product innovation with instant collaboration and live product intelligence.	PLM-DC	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
Internet of Things	Develop, customize, and operate IoT business applications in the cloud.	BC-NEO-SVC-IOT	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	Available	
Invoice Object Recommendation	Recommendation of the G/L accounts using machine learning.	CA-ML-AR-GL	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
Peppol Exchange	Meet compliance requirements by exchanging documents with the Peppol network.	LOD-LH-DCS-PAP	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)*		Available	

Service Name	Short Description	Support Component					Available as Trial	Release Status
		Environment	Capability	Infrastructure	Region			
Java Server	Develop and run Java Web applications on SAP Cloud Platform Neo Environment.	BC-NEO-RT-JAV	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)*	Yes	Available
						Europe (Frankfurt)		
						Europe (Amsterdam)		
						US East (Ashburn)		
						US West (Chandler)		
						US East (Sterling)		
						US West (Colorado Springs )		
						Australia (Sydney)		
						Japan (Tokyo)		
						Canada (Toronto)		
						Russia (Moscow)		
						Brazil (São Paulo)		
						UAE (Dubai)		
						KSA (Riyadh )		

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Job Scheduler	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-XS-SRV-JBS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available
Job Scheduler	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-XS-SRV-JBS	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Job Scheduler	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-XS-SRV-JBS	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Job Scheduler	Define and manage your jobs or Cloud Foundry tasks that run on one-time or recurring schedules.	BC-XS-SRV-JBS	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Keystore Service	Manage crypto-graphic keys and certificates.	BC-NEO-SEC-CPG	Neo	Exten-sion Suite - Devel-opment Effi-cency	SAP	Europe (Rot)* Europe (Frank-furt) Europe (Am-sterdam) US East (Ash-burn) US West (Chan-dler) US East (Ster-ling) US West (Colo-rado Springs ) Australia (Syd-neyn) Japan (Tokyo) Canada (Tor-onto) Russia (Mos-cow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh )	Yes	Available

Service Name	Short Description	Support			Region	Available as Trial	Release Status	
		Component	Environment	Capability				
Java Apps Lifecycle Management	Manage the lifecycle of Java applications by using a REST API.	BC-NEO-INFRA	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support				Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure		
SAP Live Link 365 for Authentication	Generate and validate configurable one-time PINs or verification codes.	BC-IAM-IDS		Extension Suite - Digital Experience	SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.	Available
SAP Live Link 365 for SMS	Send and receive SMSs globally via REST APIs. View traffic logs and analytics.	CEC-DI-INE		Extension Suite - Digital Experience	SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.	Available
Rapid Application Development by Mendix	Develop SAP business applications with a low-code, graphical toolset.	XX-PART-MDX-RAD		Extension Suite - Digital Experience	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	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
MongoDB	Implement a NoSQL document store.	BC-NEO-BS-MONGO	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available
MongoDB	Implement a NoSQL document store.	BC-NEO-BS-MONGO	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
MongoDB	Implement a NoSQL document store.	BC-NEO-BS-MONGO	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available



Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
MultiChain	Create MultiChain nodes and connect them to a blockchain network.	BC-BCS-MC	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	Available	

Service Name	Short Description	Support				Region	Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure			
OAuth 2.0	Protect applications and APIs with OAuth 2.0.	BC-NEO-SEC-IAM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Component	Support		Infrastructure	Region	Available as Trial	Release Status
			Environment	Capability				
Object Store	Supports storage and management of unstructured data (files, BLOBs).	BC-CP-CF-OSAAS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal) US East (VA)	Yes	Available
Object Store	Supports storage and management of unstructured data (files, BLOBs).	BC-CP-CF-OSAAS	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA) Singapore	Yes	Available
Object Store	Supports storage and management of unstructured data (files, BLOBs).	BC-CP-CF-OSAAS	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
OData Provisioning	Access data in SAP Business Suite using OData services.	OPU-GW-OD-FW	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo)	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
OData Provisioning	Access data in SAP Business Suite using OData services.	OPU-GW-OD-FW	Cloud Foundry	Extension Suite - Development Efficiency	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	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo)	Yes	Available	
Open Connectors	Simplify integration via APIs	LOD-OCN-OPS	Neo	Integration Suite	SAP	Europe (Rot)* Europe (Frankfurt) Europe (Amsterdam) US West (Chandler) US West (Colorado Springs )	Yes	Available	

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
<a href="#">Open Connectors</a>	Simplify integration via APIs	LOD-OCN-OPS	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt)	Yes	Available
						US East (VA)		
						Australia (Sydney)		
						Singapore		
						Canada (Montreal)		
						Brazil (São Paulo)		
						Japan (Tokyo)		
<a href="#">Open Connectors</a>	Simplify integration via APIs	LOD-OCN-OPS	Cloud Foundry	Integration Suite	Azure	Europe (Netherlands)	Yes	Available
						US West (WA)		
						US East (VA)		
						Singapore		
						Japan (Tokyo)		
<a href="#">SAP S/4HANA Cloud for Intelligent Product Selection</a>	Enable an intelligent customer experience for complex configurable products	LOD-PCI	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Platform Identity Provider	Use your user base from the identity authentication tenant for admin tasks.	BC-NEO-SEC-IAM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)		Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-NEO-OPS	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs) ) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Russia (Moscow) Brazil (São Paulo) Europe (Amsterdam) UAE (Dubai) KSA (Riyadh) )	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component	Environment						
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	US East (VA)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Brazil (São Paulo)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Japan (Tokyo)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Australia (Sydney)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Singapore	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	GCP	US Central (IA)	Yes	Available	
Portal	Create role based, multi-channel sites to access business apps and content.	EP-CPP-CF-OPS	Cloud Foundry	Extension Suite - Digital Experience	Alibaba	China (Shanghai)**	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POST-GRES	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POST-GRES	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
PostgreSQL	Consume an object-relational database with PostgreSQL.	BC-NEO-BS-POST-GRES	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
Pricing service	Calculate prices for configurable and non-configurable products	LOD-CPS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Japan (Tokyo) Brazil (São Paulo)	Yes	Available
Process Visibility	Cloud Offering with End-to-End visibility on Business Processes	LOD-BPM-VIS	Cloud Foundry	Extension Suite - Digital Process Automation	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
Variant Configuration service	Configure your SAP ERP or SAP S/4HANA products interactively in the cloud	LOD-CPS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Singapore	Yes	Available

Service Name	Short Description	Support				Region	Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure			
Java Profiling	Profile and analyze your Java applications.	BC-JVM Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh )	Yes	Available	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Quorum	Create Quorum nodes and connect them to a blockchain network.	BC-BCS-QRM	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	Available	
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RAB-BITMQ	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available	
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RAB-BITMQ	Cloud Foundry	Integration Suite	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available	
RabbitMQ	Get robust asynchronous messaging between applications.	BC-NEO-BS-RAB-BITMQ	Cloud Foundry	Integration Suite	GCP	US Central (IA)	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-RE-DIS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
						US East (VA)			
						Brazil (São Paulo)			
						Japan (Tokyo)			
						Australia (Sydney)			
						Singapore			
						Canada (Montreal)			
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-RE-DIS	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
						US West (WA)			
						US East (VA)			
						Singapore			
						Japan (Tokyo)			
Redis	Implement an in-memory caching layer with Redis.	BC-NEO-BS-RE-DIS	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Remote Data Sync	Synchronize data between remote databases and a cloud SAP HANA database.	BC-NEO-CON	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo)	Yes	Available
Data Retention Manager	Manage retention and residence rules to block or delete personal data.	LOD-GDP-RM	Cloud Foundry	Extension Suite - Development Efficiency	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	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure		Available as Trial	Release Status
					Region			
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-WNG	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Canada (Montreal) Japan (Tokyo) Brazil (São Paulo) Singapore	Yes	Available
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-WNG	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore	Yes	Available
SAP Business Application Studio	Develop, debug, test, and deploy SAP business applications.	CA-WNG	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support			Available as Trial	Release Status		
		Component	Environment	Capability				
Launchpad	Simplify access to applications by establishing a central launchpad.	EP-CPP-CF	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore	Yes	Available
Launchpad	Simplify access to applications by establishing a central launchpad.	EP-CPP-CF	Cloud Foundry	Extension Suite - Digital Experience	Azure	US West (WA) US East (VA) Europe (Netherlands) Japan (Tokyo) Singapore	Yes	Available

Service Name	Short Description	Support						
		Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
Document Management, application option	Organize your documents with ready-to-use document management capabilities.	BC-CP-CF-SDM	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal)	Yes	Available
Document Management, application option	Organize your documents with ready-to-use document management capabilities.	BC-CP-CF-SDM	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) US East (VA) Japan (Tokyo) Singapore	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt) Australia (Sydney) US East (VA) Singapore Japan (Tokyo) Brazil (São Paulo)	Yes	Available
Workflow Management	Digitize workflows, manage decisions and gain end-to-end process visibility	LOD-BPM-PFS	Cloud Foundry	Extension Suite - Digital Process Automation	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	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	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
SAP API Business Hub	Discover, explore and test the APIs offered by SAP.	LOD-CHB		Integration Suite	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			Available as Trial	Release Status		
		Component	Environment	Capability				
SAP ASE Service	Create and consume SAP ASE databases.	BC-NEO-PERS	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
SAP Blockchain Business Services	Use blockchain capabilities to enhance your cross-company integration.	BC-BCS-BBS	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA)	Yes	BETA	
SAP Analytics Cloud	Carry out business intelligence, planning, and predictive analysis tasks.	LOD-ANA	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) Europe (Frankfurt) UAE (Dubai) KSA (Riyadh)	Available		

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
SAP Analytics Cloud	Carry out business intelligence, planning, and predictive analysis tasks.	LOD-ANA	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore	Yes	Available	
SAP Analytics Cloud	Carry out business intelligence, planning, and predictive analysis tasks.	LOD-ANA	Cloud Foundry	Extension Suite - Digital Experience	Alibaba	China (Shanghai)**	Yes	Available	
Application Runtime	Operate polyglot cloud applications in Cloud Foundry on the SAP Cloud Platform.	BC-CP-CF	Cloud Foundry	Extension Suite - Development Efficiency	AWS	US East (VA) Europe (Frankfurt) Brazil (São Paulo) Japan (Tokyo) Canada (Montreal) Australia (Sydney) Singapore	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component	Environment						
Application Runtime	Operate polyglot cloud applications in Cloud Foundry on the SAP Cloud Platform.	BC-CP-CF	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
Application Runtime	Operate polyglot cloud applications in Cloud Foundry on the SAP Cloud Platform.	BC-CP-CF	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
Application Runtime	Operate polyglot cloud applications in Cloud Foundry on the SAP Cloud Platform.	BC-CP-CF	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available	
Functions	Create functions using the principles of serverless computing.	OPU-GW-OD-FUN	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	BETA	
						US East (VA)			

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	Integration Suite	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	Integration Suite	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
Integration Suite	Integrate applications, services, and systems across landscapes.	LOD-HCI-PI	Cloud Foundry	Integration Suite	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
SAP Conversational AI	Build and deploy innovative chatbots using this comprehensive end-to-end platform.	CA-ML-RAI	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt)	Yes	Available	
SAP Customer Order Sourcing	Calculate sourcing results based on your own sourcing strategies.	LOD-CID-OSR	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
SAP Data Intelligence	Orchestrate, refine, enrich, apply intelligence on your governed data across your entire distributed data landscape.	CA-DI	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Japan (Tokyo)	Yes	Available	
SAP Data Intelligence	Orchestrate, refine, enrich, apply intelligence on your governed data across your entire distributed data landscape.	CA-DI	Cloud Foundry	Integration Suite	Azure	Europe (Netherlands) US West (WA)	Yes	Available	

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
SAP Document Center	Use uniform standard-based file access and mobilize your business content.	BC-NEO-ECM-APP	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) US West (Colorado Springs ) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh )	Yes	Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
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	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
SAP HANA Cloud	A single gateway to all your data.	HAN-CLS-HC	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Australia (Sydney) Singapore Brazil (São Paulo) Canada (Montreal) Europe (Frankfurt) Japan (Tokyo) US East (VA)	Yes	Available	
SAP HANA Cloud	A single gateway to all your data.	HAN-CLS-HC	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Singapore Europe (Netherlands) Japan (Tokyo) US West (WA) US East (VA)	Yes	Available	

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
SAP HANA spatial services	Build spatially-aware business applications across all business processes.	BC-CP-CF-HSS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
SAP HANA Service	Create and consume SAP HANA databases.	BC-NEO-PERS	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh )	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Support		Available as Trial	Release Status
					Infrastructure	Region		
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available
SAP HANA Service	Create and consume SAP HANA databases.	HAN-CLS-DB	Cloud Foundry	Extension Suite - Development Efficiency	Alibaba	China (Shanghai)**	Yes	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
SAP Intelligent Robotic Process Automation	Design, configure, and execute automation projects.	CA-ML-IPA	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt) 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	Extension Suite - Digital Process Automation	Alibaba	China (Shanghai)**	Yes	Available
SAP IoT	Put raw sensor data into business context and leverage it in analytical or transactional applications.	IOT-BSV-APB	Cloud Foundry	Integration Suite	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	Integration Suite	Azure	Europe (Netherlands)	Yes	Available

Service Name	Short Description	Support			Available as Trial	Release Status		
		Component	Environment	Capability	Infrastructure			
SAP IoT Connect 365	Simplify the complex connectivity, scalability, and management of IoT.	BC-NEO-SVC-IOT		Extension Suite - Development Efficiency	SAP	This service does not run on standard SAP Cloud Platform regions. Check out <a href="#">this note</a> for further details.		
SAP Leonardo ML Foundation	Infuse your applications with intelligent, easy-to-use services based on Machine Learning.	CA-ML-PLT	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) 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	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available
Market Rates, Bring Your Own Rates	Upload market rates and download the same multiple times from different systems.	LOD-CBS-CS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support			Region	Available as Trial	Release Status	
		Component	Environment	Capability				
Market Rates, Refinitiv	Get daily and historical Exchange Rates and Interest Rates from Refinitiv.	LOD-CBS-CS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available
SAP Omni-channel Promotion Pricing	Calculate effective sales prices by applying promotional rules.	LOD-CID-OPP	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available
SAP Process Mining by Celonis, cloud edition	Mine data from your operational systems, visualize processes, and transform your business with AI.	XX-PART-CEL	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt)	Yes	Available
SAP Procurement Intelligence	Automate and simplify your business processes by applying machine learning intelligence.	CA-ML-PA	Cloud Foundry	Integration Suite	AWS	Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component	Environment						
SAP Real-Spend	Manage and track expenses, and compare them against your budget in real time.	LOD-MAP-RS	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Frankfurt) US East (Ashburn) US East (Sterling) US West (Colorado Springs)	Yes	Available	
SAP S/4HANA Cloud Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP S/4HANA Cloud system.	BC-NEO-EXT-S4C	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Brazil (São Paulo) Japan (Tokyo) Australia (Sydney) Singapore Canada (Montreal)	Yes	Available	

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
SAP S/4HANA Cloud Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP S/4HANA Cloud system.	BC-NEO-EXT-S4C	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
SAP S/4HANA Cloud Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP S/4HANA Cloud system.	BC-NEO-EXT-S4C	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
SAP SuccessFactors Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP SuccessFactors system.	BC-NEO-EXT-SF	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Brazil (São Paulo)	Yes	Available	
					AWS	Australia (Sydney)			
					AWS	Europe (Frankfurt)			
					AWS	US East (VA)			
					AWS	Canada (Montreal)			
					AWS	Singapore			
					AWS	Japan (Tokyo)			

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
SAP SuccessFactors Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP SuccessFactors system.	BC-NEO-EXT-SF	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo)	Yes	Available
SAP SuccessFactors Extensibility	Connects extension applications running in an SAP Cloud Platform subaccount to an SAP SuccessFactors system.	BC-NEO-EXT-SF	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
SAP Translation Hub	Translate UI texts and get suggestions for UI texts during development.	LOD-TH	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Europe (Amsterdam) US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
SAP Web IDE Full-Stack	Create and extend SAP full-stack applications for browsers and mobile devices.	CA-WDE-PLFRM	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Service Management	The central registry for service brokers and platforms in SAP Cloud Platform.	BC-NEO-SVCMGR	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
Service Management	The central registry for service brokers and platforms in SAP Cloud Platform.	BC-NEO-SVCMGR	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands)	Yes	Available	
Service Management	The central registry for service brokers and platforms in SAP Cloud Platform.	BC-NEO-SVCMGR	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	Yes	Available	
Service Ticket Intelligence	Build a self-driven customer service powered by machine learning.	CA-ML-STI	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt)	Yes	Available	
						US East (VA)			

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Service Ticket Intelligence	Build a self-driven customer service powered by machine learning.	CA-ML-STI	Cloud Foundry	Extension Suite - Development Efficiency	GCP	US Central (IA)	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	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)*	Yes	Available	

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
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	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA)	Yes	Available

Service Name	Short Description	Support		Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
		Component							
Solutions Lifecycle Management	Deploy, subscribe and transport solutions using multi-target applications (MTAs).	BC-NEO-LCM-SRV	Neo	Extension Suite - Development Efficiency	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Streaming Analytics	Process continuous streams of event data in real time and act on the results.	HAN-SDS	Neo	Extension Suite - Digital Experience	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) US East (Sterling) Australia (Sydney) Japan (Tokyo) Canada (Toronto) Brazil (São Paulo) KSA (Riyadh ) Russia (Moscow) Europe (Amsterdam) Europe (Frankfurt) UAE (Dubai)	Yes	Available
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* US West (Chandler) US East (Sterling)	Yes	Available

Service Name	Short Description	Support						
		Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA)	Yes	Available
Tax Service	Determine and calculate indirect taxes to support tax compliance.	LOD-LH-TAX	Cloud Foundry	Extension Suite - Development Efficiency	Azure	US West (WA)	Yes	Available
Transport Management	Manage transports of development artifacts and application-specific content.	BC-CP-LCM-TMS	Cloud Foundry	Extension Suite - Development Efficiency	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Canada (Montreal) Brazil (São Paulo) Japan (Tokyo)	Yes	Available
Transport Management	Manage transports of development artifacts and application-specific content.	BC-CP-LCM-TMS	Cloud Foundry	Extension Suite - Development Efficiency	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available

Service Name	Short Description	Support			Infrastructure	Region	Available as Trial	Release Status
		Component	Environment	Capability				
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Neo	Extension Suite - Digital Experience	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) Russia (Moscow) Brazil (São Paulo) UAE (Dubai) KSA (Riyadh)	Yes	Available

Service Name	Short Description	Component	Support			Region	Available as Trial	Release Status
			Environment	Capability	Infrastructure			
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	Extension Suite - Digital Experience	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	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) Japan (Tokyo) Singapore US East (VA)	Yes	Available
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	Extension Suite - Digital Experience	GCP	US Central (IA)	Yes	Available

Service Name	Short Description	Support			Available as Trial	Release Status	
		Component	Environment	Capability			
UI Theme Designer	Apply your corporate branding to applications based on SAPUI5 technology.	CA-U12-THD	Cloud Foundry	Extension Suite - Digital Experience	Alibaba	China (Shanghai)**	Yes Available
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-U15-FL-CLS	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Japan (Tokyo) Singapore Australia (Sydney) Brazil (São Paulo)	Yes Available
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-U15-FL-CLS	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes Available
UI5 flexibility for key users	Add UI adaptation to your UI5 applications.	CA-U15-FL-CLS	Cloud Foundry	Extension Suite - Digital Experience	GCP	US Central (IA)	Yes Available

Service Name	Short Description	Support						Available as Trial	Release Status
		Component	Environment	Capability	Infrastructure	Region			
Virtual Machines	Virtualized hardware resources (CPU, RAM, disk space, installed OS) to install and maintain the Linux-based software.	BC-NEO-INFRA-VM	Neo	Extension Suite - Development Efficiency	SAP	Europe (Rot)* Australia (Sydney) US East (Sterling)		Available	
Web Analytics	Analyze usage of your websites and web applications.	CA-SWA	Cloud Foundry	Extension Suite - Digital Experience	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Brazil (São Paulo) Japan (Tokyo) Singapore	Yes	Available	
Web Analytics	Analyze usage of your websites and web applications.	CA-SWA	Cloud Foundry	Extension Suite - Digital Experience	Azure	Europe (Netherlands) US West (WA) US East (VA) Singapore Japan (Tokyo)	Yes	Available	

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Neo	Extension Suite - Digital Process Automation	SAP	Europe (Rot)* US East (Ashburn) US West (Chandler) Australia (Sydney) Japan (Tokyo) US East (Sterling) US West (Colorado Springs) Canada (Toronto) Russia (Moscow) Brazil (São Paulo) Europe (Amsterdam) UAE (Dubai) KSA (Riyadh) Europe (Frankfurt)	Yes	Available

Service Name	Short Description	Component	Environment	Capability	Infrastructure	Region	Available as Trial	Release Status
							Support	
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	Extension Suite - Digital Process Automation	AWS	Europe (Frankfurt) US East (VA) Australia (Sydney) Singapore Japan (Tokyo) Brazil (São Paulo)	Yes	Available
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	Extension Suite - Digital Process Automation	Azure	Singapore Japan (Tokyo) Europe (Netherlands) US West (WA) US East (VA)	Yes	Available
Workflow	Automate business processes using workflow technology.	LOD-BPM-WFS	Cloud Foundry	Extension Suite - Digital Process Automation	Alibaba	China (Shanghai)**	Yes	Available

## Additional Components

### Tools

Name	Support Component
SAP Cloud Platform cockpit	BC-NEO-CPT (Neo environment) BC-CP-CF-CPT (Cloud Foundry environment)
CLI for SAP Cloud Platform	BC-CP-TOOLS-CLI
Eclipse Java tools	BC-NEO-ECL-JAV
SAP Cloud Platform console client	BC-NEO-CMDTOOL
SAP UI Development Toolkit for HTML5 (SAPUI5)	CA-WDE
SAP Cloud Platform Software Development Kit	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, BC-NEO-INFR
Commercial Infrastructure	BC-NEO-CIS
Metering Service	BC-NEO-MET
Audit Log	BC-NEO-AUDITLOG (Neo environment) BC-CP-CF-SEC-AUDITLG (Cloud Foundry environment)
SAP Cloud Platform 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)

# 8 Glossary

## SAP Cloud Platform Terminology

### A-G

application	Software hosted on SAP Cloud Platform 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 77]</a>	The single point-of-entry for an application running in the Cloud Foundry environment on SAP Cloud Platform. 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="#">buildpack</a>	In the Cloud Foundry environment, buildpacks provide framework and runtime support for apps.
<a href="#">business service [page 116]</a>	Platform services that enable, facilitate, or accelerate the development of business process components and elements of a business application.
<a href="#">cockpit</a>	SAP Cloud Platform cockpit is the central point of entry to key information about your accounts and applications, and for managing all activities associated with your account.
<a href="#">connectivity</a>	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 Cloud Platform and existing on-premise systems. It combines an easy setup with a clear configuration of the systems that are exposed to SAP Cloud Platform.

Cloud Controller API	Manages the lifecycle of applications. When a developer pushes an application to the Cloud Foundry environment, this is targeting the Cloud Controller. The Cloud Controller then stores the raw application bits, creates a record to track the application metadata, and directs a DEA node to stage and run the application. The Cloud Controller also maintains records of orgs, spaces, services, service instances, user roles, and more.
cloud management tools	<p>Cloud management tools represent the group of technologies designed for managing SAP Cloud Platform.</p> <p>Cloud management tools is a synonym for the internally used term Foundation.</p>
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</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.
environment	Constitutes the SAP Cloud Platform actual Platform-as-a-Service offering that allows for the development and administration of business 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.
global account	<p>Contains the entitlements of a customer. A global account exists independently of its usage in a concrete environment and across all regions. A global account can be either a trial account or an enterprise account and allows for member and quota management. For more information, see <a href="#">Account Model</a>.</p> <p>Contains one or more subaccounts and the entitlements of a customer. A global account allows for member and quota management.</p>

## H-R

HDI	The HANA Deployment Infrastructure (HDI) provides a service layer on top of the SAP HANA database to deploy database artifacts into it.
-----	---

HDI container	A set of schemas and users that together enable an isolated deployment of SAP HANA database artifacts. The next-generation SAP HANA Extended Application Services (XS) provide a service to create HDI containers on a shared database, and a mechanism to deploy database artifacts together with the application code. See "service broker".
identity provider (IdP)	An authorization authority containing all user information and credentials. In SAP Cloud Platform, user information is provided by identity providers, not stored in SAP Cloud Platform itself.
managed service	Services that integrate with the Cloud Foundry environment using a service broker that implements the Service Broker API. Managed services enable end users to provision reserved resources and credentials on demand.
member	Indicates a user's assignment to an account. As an account member, a user automatically has the permissions required to use the SAP Cloud Platform functionality within the scope of the respective account and as permitted by their account member roles.
multi-cloud foundation	SAP's platform-as-a-service offering that provides application development services and capabilities on multiple cloud infrastructure providers. The multi-cloud foundation supports multiple environments, such as Cloud Foundry, ABAP, and Kyma.
<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.
org (Organization)	A hierarchical level in the account structure of SAP Cloud Platform 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 business applications and other platform services on SAP Cloud Platform. Platform services are integrated with business applications and other cloud resources by developers. End users only interact with platform services via business 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 business applications on SAP Cloud Platform. For example, a programming model can include programming languages, runtimes, and APIs.
quota	A numeric quantity that defines the maximum allowed consumption of a specific technical asset/resource.
recovery point objective (RPO)	The time between the most recent backed up data and the disaster. Any data that has already been backed up must not be lost.
recovery time objective (RTO)	The time between the disaster and the restoration of the affected service. The time needed for declaring the disaster is included in the RTO.
region	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

SAP ID service	The default identity provider for SAP Cloud Platform 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.
SAP Cloud Platform , Cloud Foundry environment	An open Platform-as-a-Service, which provides a scalable runtime container and a choice of clouds, runtimes, and services.
SAP Cloud Platform, Neo environment	An enterprise Platform-as-a-Service, providing a range of services to our customers
SAPUI5	A development toolkit providing UI controls for developing Web applications.
service broker	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.
service plan	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.
service provider	The application interested in getting authentication and authorization information. Instead of providing this information in itself, it contacts the identity provider.
service rate	A fixed monetary amount charged for the consumption of a service plan offered on SAP Cloud Platform. For example, each t-shirt size of a database has a specific service rate attached to it.
service rate plan	A formula for translating usage data of a service into a monetary amount. A service rate plan can comprise a one-time setup fee, a recurring monthly rate, or block rates that describe a metric. A metric can be any type of quantity that can be measured.
<a href="#">SAP Java Virtual Machine</a>	SAP's own implementation of a Java Virtual Machine on which the SAP Cloud Platform infrastructure runs.
space	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.

subaccount	Lets you structure a global account according to customer requirements with regards to members, authorizations and quotas.  In the Neo environment, an enterprise account can have one or many subaccounts, while a trial account can have only one. Neo apps and services run only in the Neo environment, however, HTTPS-based services might be cross-consumed from the outside.  In the Cloud Foundry environment, a global account can have as many subaccounts as required; each subaccount can have an associated Cloud Foundry org.
subscription	In the context of SAP Cloud Platform, "subscription" means either of the following:  1. The SAP Cloud Platform current business model, which is a pre-paid monthly recurring charge for the right to use a bundle of technical assets up to specified limits.  2. A technical subscription to a business application that is either provided by SAP or by a customer in a different subaccount. Also referred to as software-as-a-service.
technical services	Platform services that enable, facilitate, or accelerate the generic development of a business application, independent of the application's business process or task.
tool	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.
Tenant ID	Identifier of the application consumer in the current application context. The tenant ID can be used to distinguish data of different application consumers.
user-provided service instance	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.
WTP server adapter	A tool for deploying and testing Java EE assets on SAP Cloud Platform or for local testing.

# 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 a 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.

## Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

