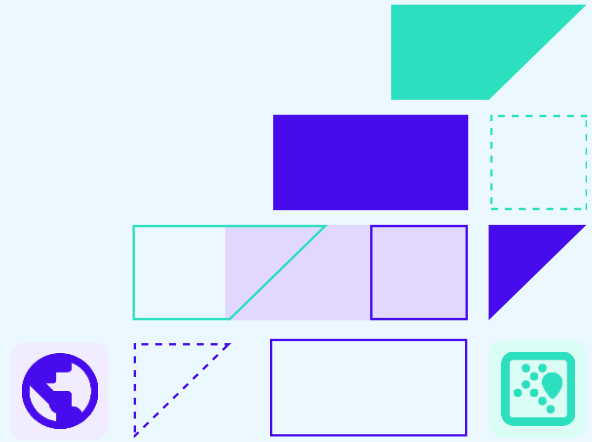
SAP TechEd

SAP TechEd

Delve into ABAP Cloud on SAP BTP, ABAP environment – AD181v

Carine Tchoutouo Djomo, Merve Temel – SAP SE
November 3, 2023



Agenda

INTRODUCTION

TODAY’S SCENARIO – OVERVIEW

TODAY’S SCENARIO – EXERCISES

WRAP UP

INTRODUCTION

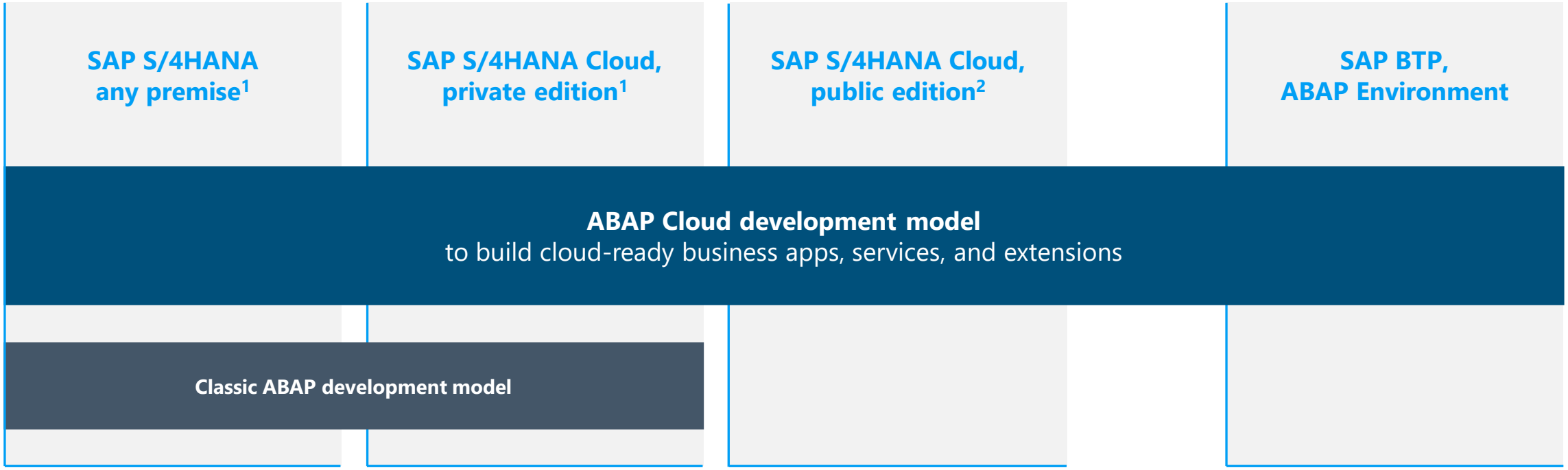
ABAP Cloud

... is the ABAP development model to build cloud-ready business apps, services, and extensions

... comes with SAP Business Technology Platform (SAP BTP) and SAP S/4HANA

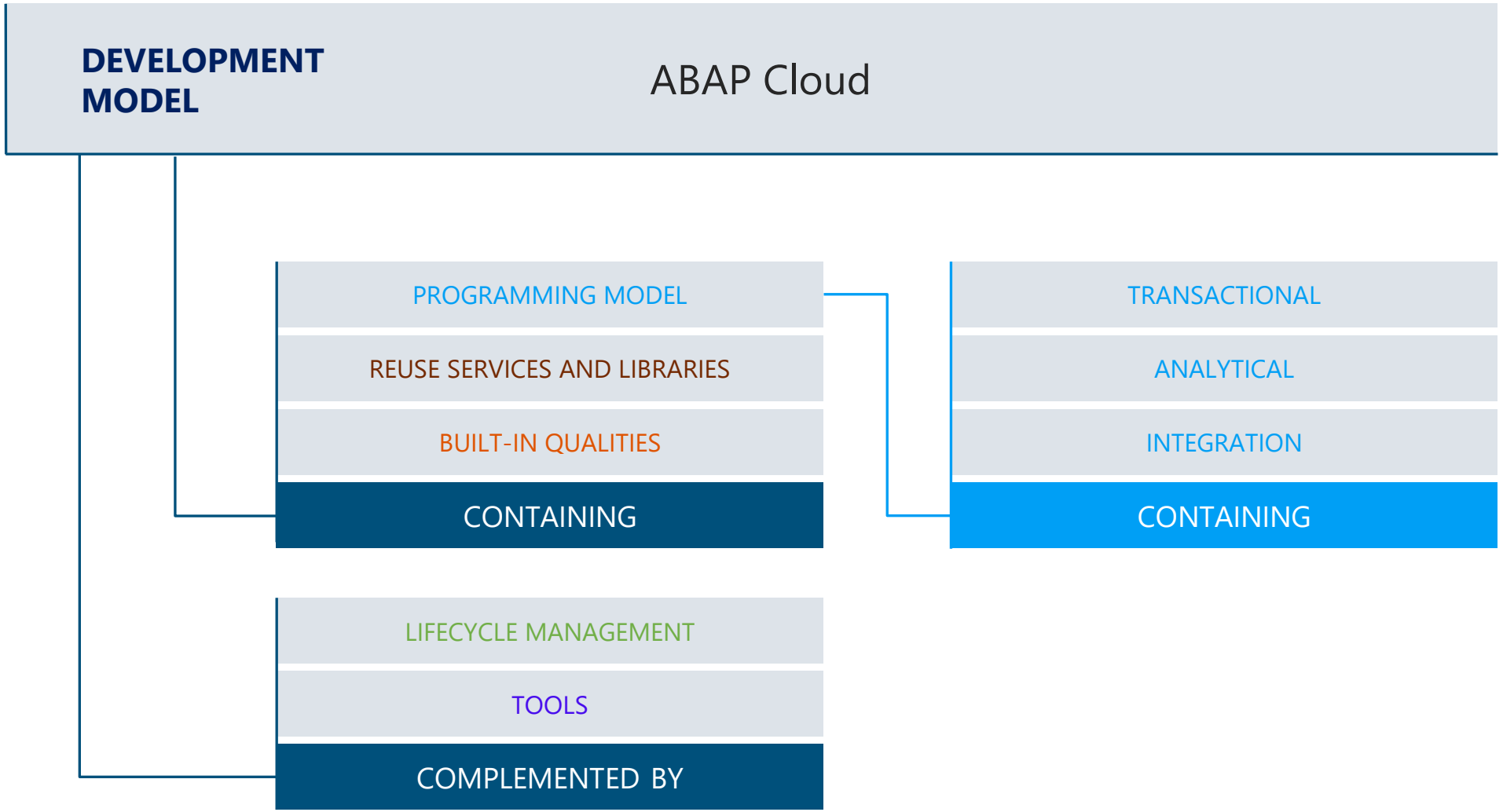
... works with public or private cloud, and even on-premise

ABAP Cloud – One development model for SAP S/4HANA and SAP BTP



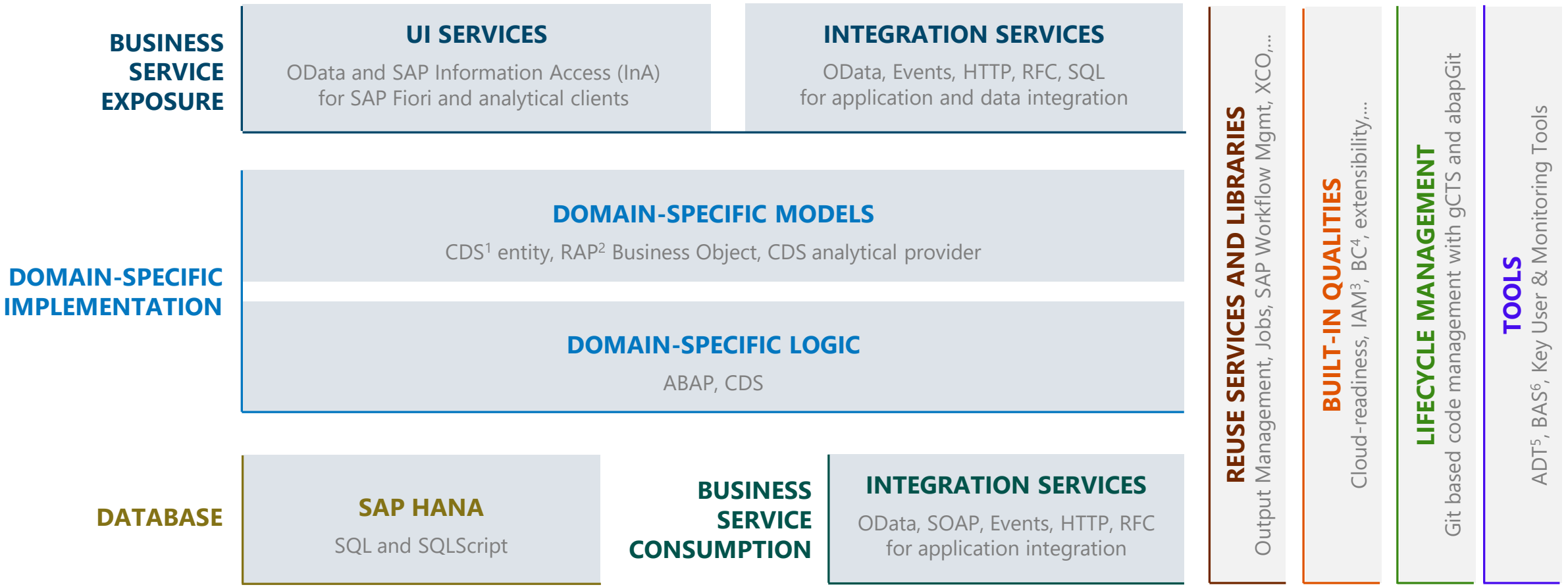
[1] SAP S/4HANA any premise or SAP S/4HANA Cloud, private edition release ≥ 2022
[2] SAP S/4HANA Cloud, public edition release ≥ 2208, 3-system landscape required

ABAP Cloud – What's in it?



Learn more: [ABAP Cloud - What does it comprise?]

ABAP Cloud map

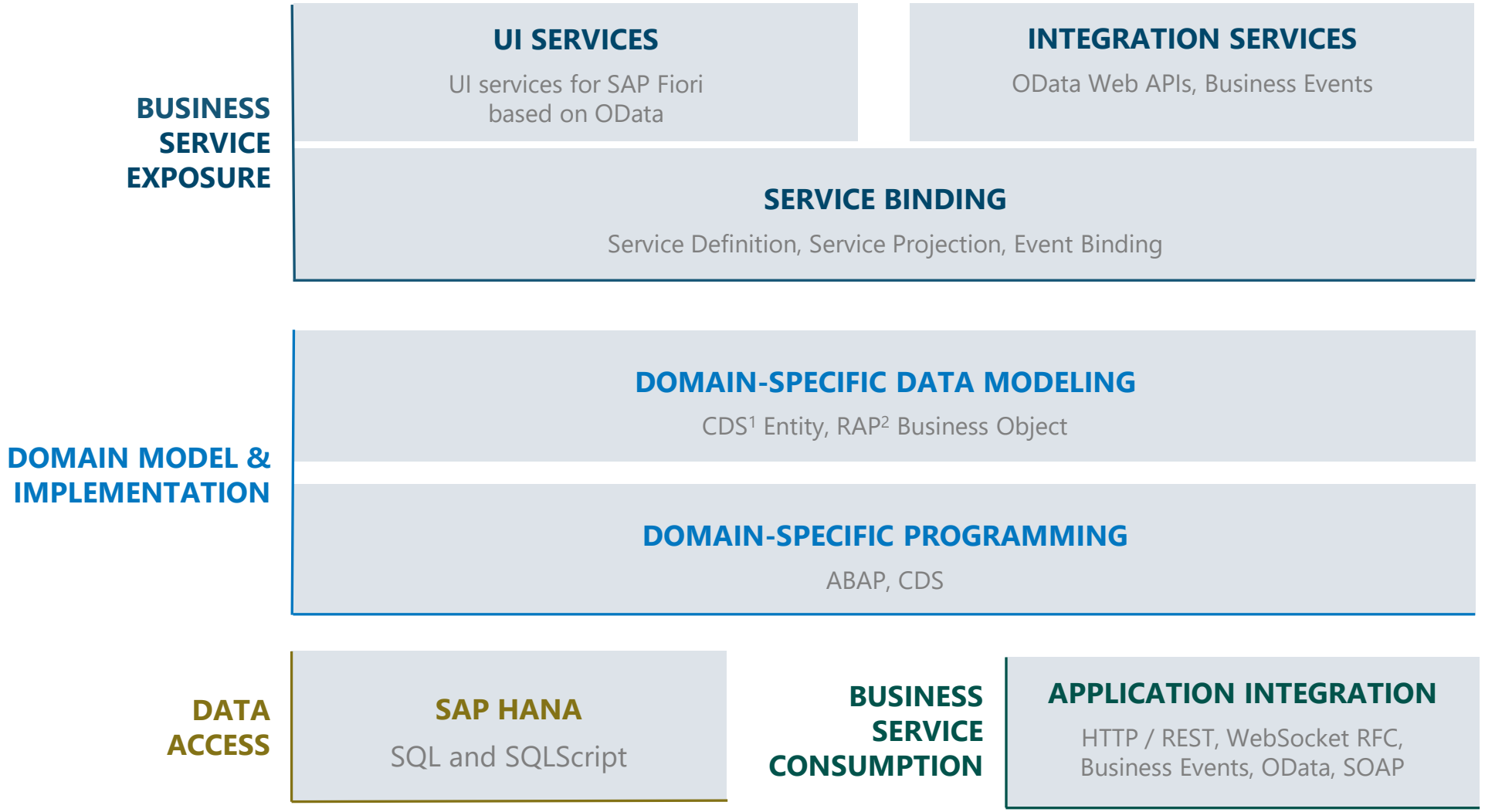


Related sessions: AD103v, AD107v, DT187v, DT182v, DT103v, DT200v

[1] Core Data Services
[2] ABAP RESTful application programming model
[3] Identity & Access Management
[4] Business Configuration
[5] ABAP Development Tools
[6] SAP Business Application Studio

ABAP Cloud – Transactional scenarios

The ABAP RESTful application programming model (RAP)

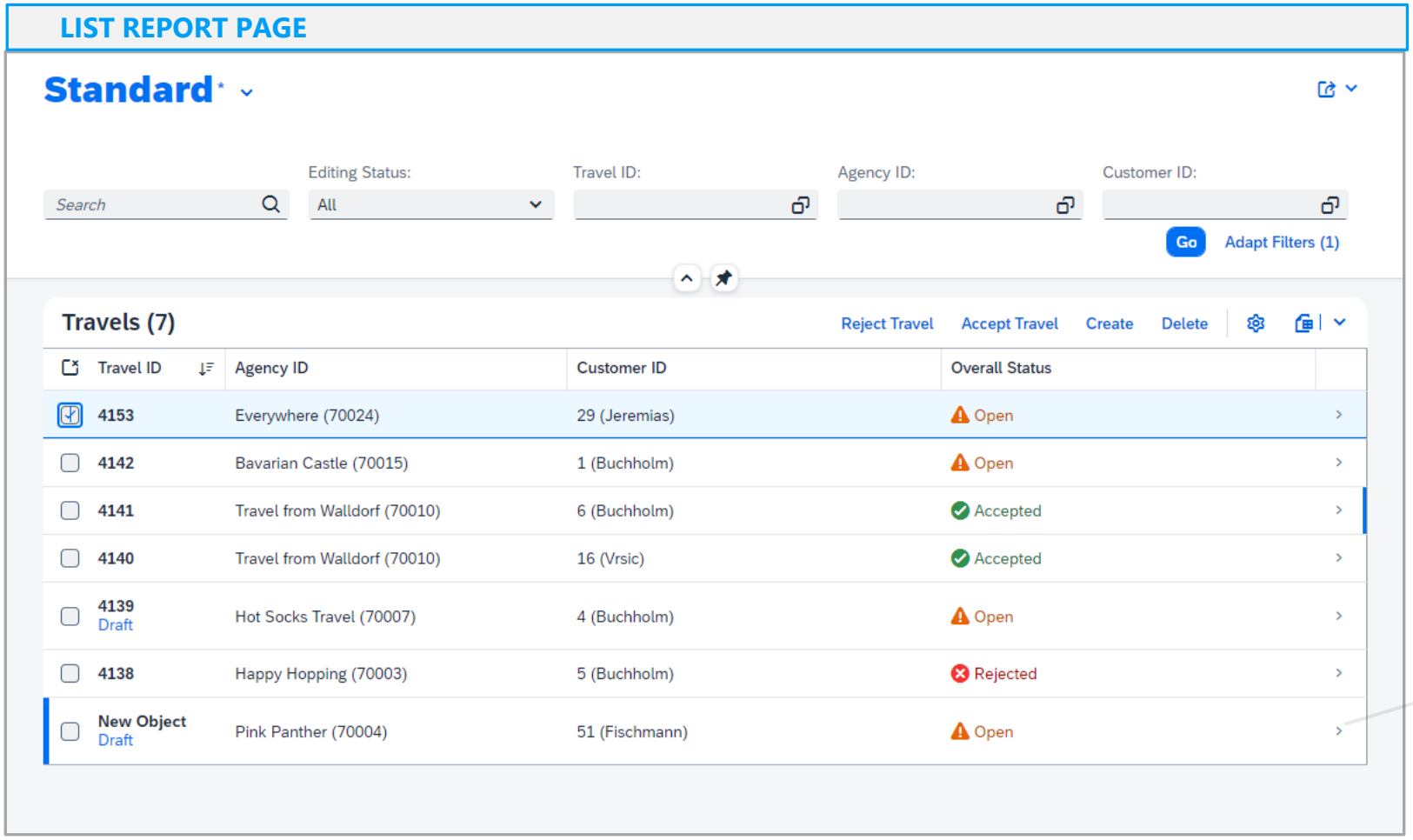


[1] Core Data Services
[2] ABAP RESTful application programming model

TODAY’S SCENARIO – OVERVIEW

Building transactional SAP Fiori apps with RAP

with search, filter, and draft capabilities



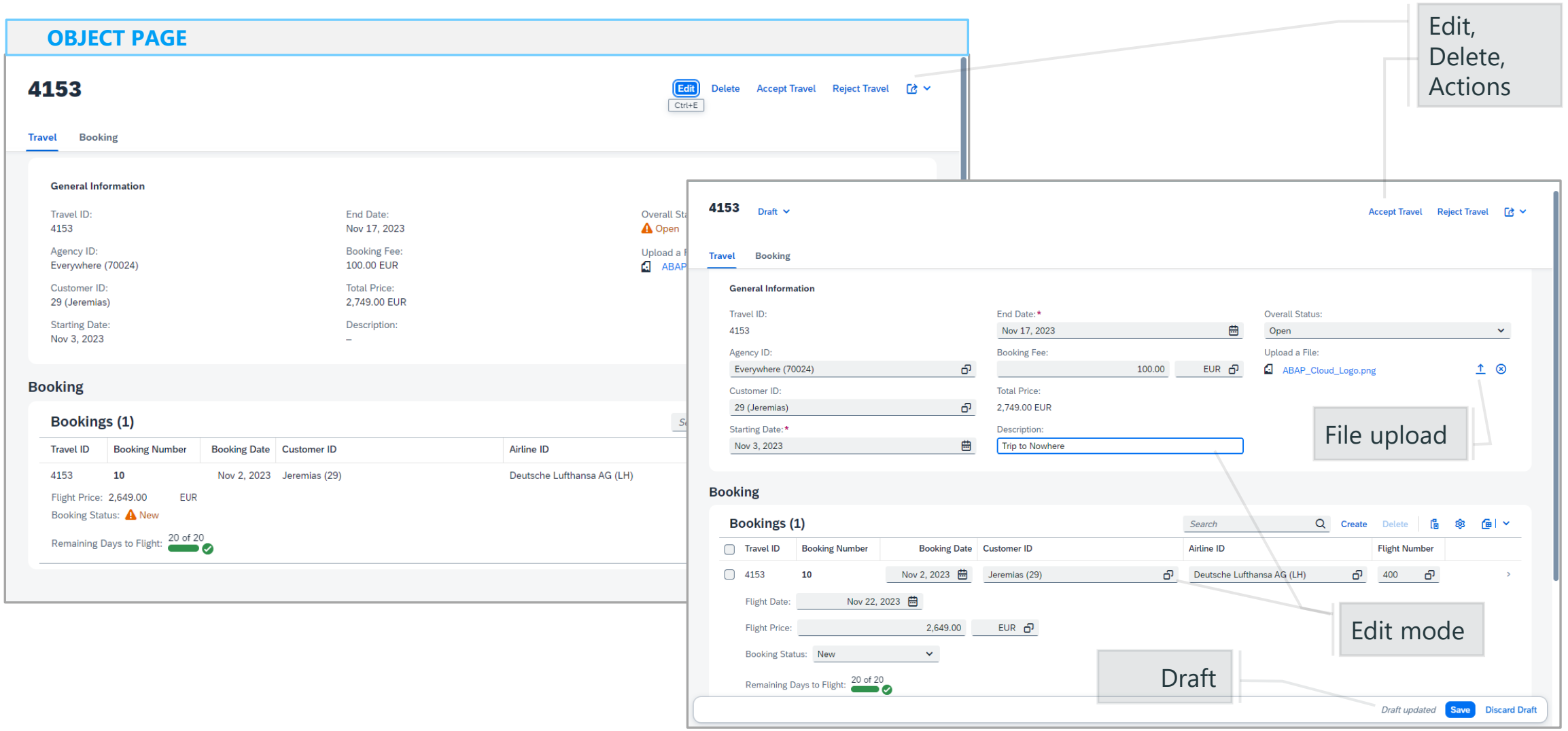
Navigation to object page

Some of the used RAP features:

- Large objects / OData streams
- Virtual elements
- Late numbering
- Internal and determine instance actions
- Static (default) factory action
- Side effects
- Functions
- Business events
- etc.

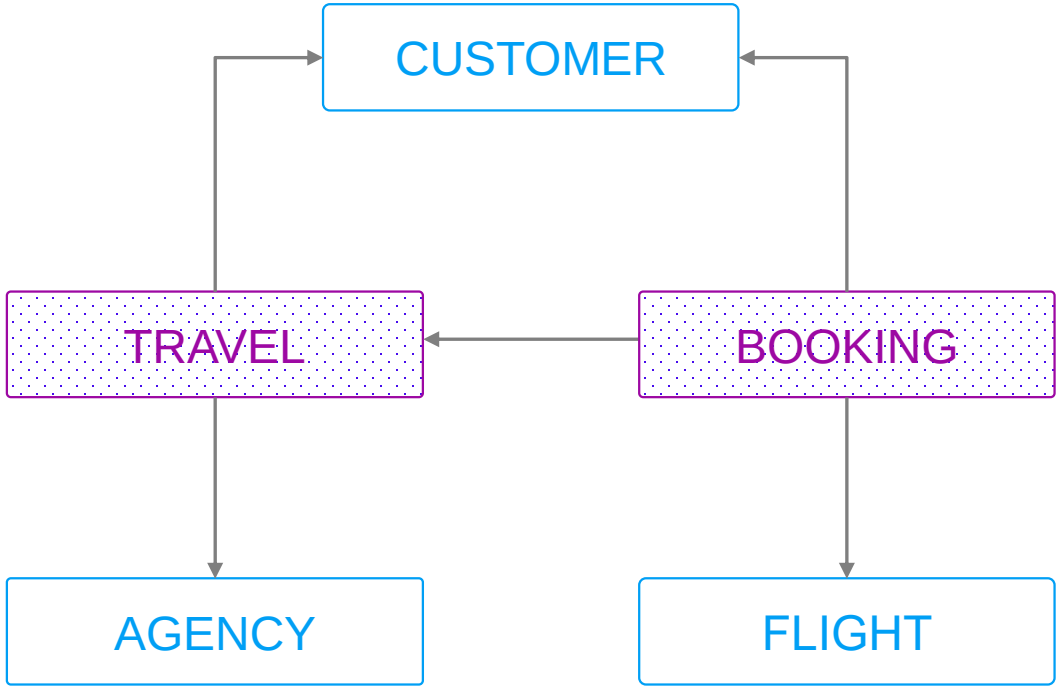
Building transactional SAP Fiori apps with RAP

Travel list report app >> Object page



Building transactional SAP Fiori apps with RAP

Underlying simplified Flight data model



A **Travel entity** defines general travel data, such as the agency ID or customer ID, the overall status of the travel booking, and the total price of the travel.

A **Booking entity** comprises general flight and booking data, the customer ID for whom the flight is booked as well as the travel ID to which the booking belongs.

A **Customer entity** provides a detailed description of a flight customer, or passenger, such as the name, the address, and contact data.

An **Agency entity** defines travel agency data, such as the address and contact data.

A **Flight entity** defines the general flight data for each connection.

DEMO

TODAY'S SCENARIO – EXERCISES

The AD181v hands-on exercises

Exercises

^Top of page

Follow these steps to enhance an existing OData UI service developed with RAP for a transactional SAP Fiori elements-based Travel Processing app.

Exercises	Boosters
Getting Started	--
Exercise 1: Inspect your Exercise Package - Generated RAP BO & UI Service	--
Exercise 2: Enhance the Data Model of the Base and Projected BO	--
Exercise 3: Enhance the BO Behavior Definition and Projection	💡
Exercise 4: Implement the Base BO Behavior - Late Numbering	--
Exercise 5: Adjust the UI Semantics in the Metadata Extensions	💡
Exercise 6: Implement the Base BO Behavior - Validations	💡
Exercise 7: Implement the Base BO Behavior - Actions	💡
Exercise 8: Implement the Base BO Behavior - Determinations	--
Exercise 9: Enhance the BO Behavior with Side Effects	--
Exercise 10: Implement the Base BO Behavior - Functions	--
Exercise 11: Enhance the BO Behavior with Business Events	--
Exercise 12: Implement the Base BO Behavior - Dynamic Feature Control	--

BO: Business Object
EML: Entity Manipulation Language
💡 Exercises with *Boosters* offer an accelerated way of doing them.

01 Access to the exercises:
https://github.com/SAP-samples/teched2023-AD181v

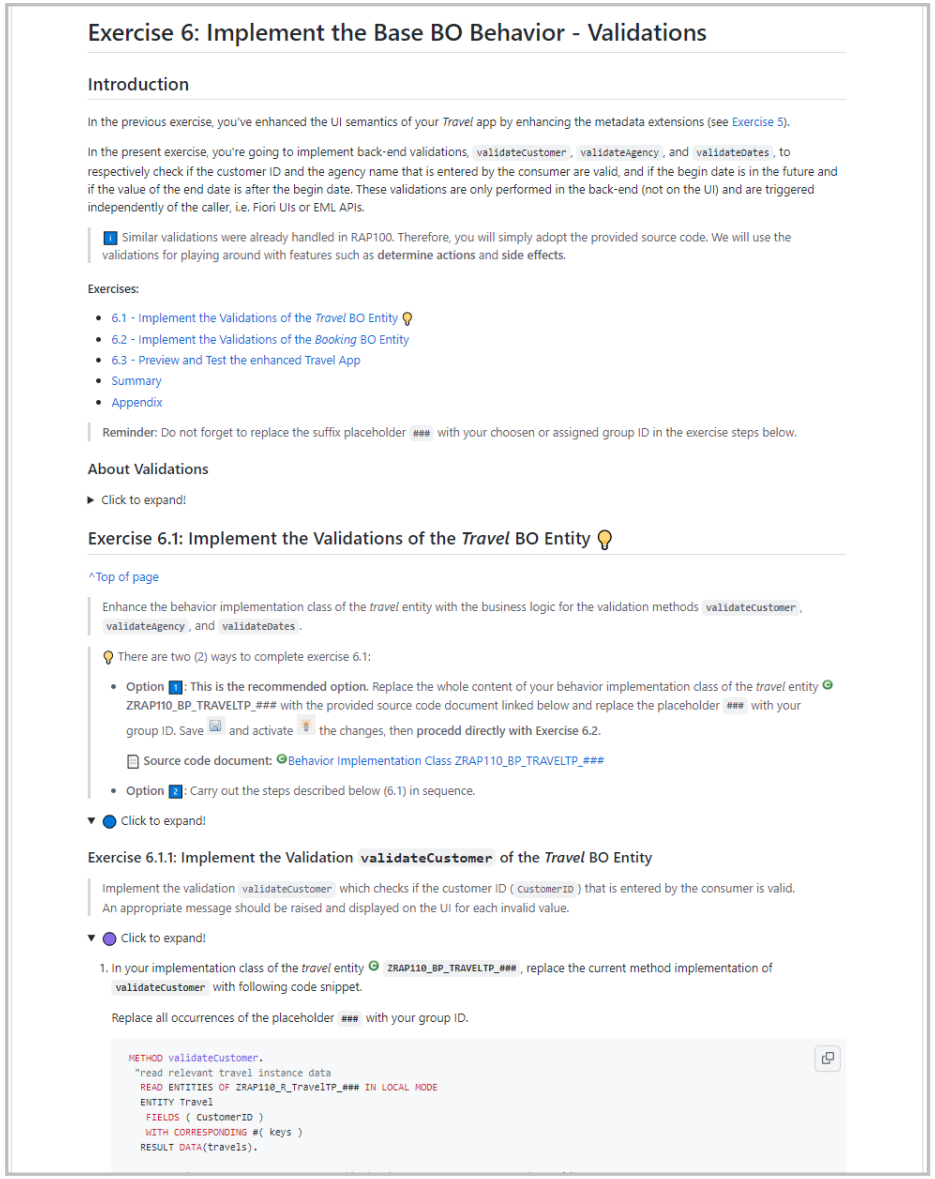
02 Set up your development environment.
1. Install and set up your ABAP Development Tools for Eclipse
2. Create an SAP BTP ABAP Environment Trial User
3. Create an ABAP Cloud Project

03 Move at your own speed!
▶ Use the provided 💡boosters if you are already familiar with a particular topic

04 Feel free to ask questions via GitHub issues!
▶ Provide screenshots to facilitate the support

The AD181v hands-on exercises – Working on GitHub (1)

Follow the step-by-step instructions provided on GitHub.



Exercise 6: Implement the Base BO Behavior - Validations

Introduction

In the previous exercise, you've enhanced the UI semantics of your *Travel* app by enhancing the metadata extensions (see Exercise 5).

In the present exercise, you're going to implement back-end validations, `validateCustomer`, `validateAgency`, and `validateDates`, to respectively check if the customer ID and the agency name that is entered by the consumer are valid, and if the begin date is in the future and if the value of the end date is after the begin date. These validations are only performed in the back-end (not on the UI) and are triggered independently of the caller, i.e. Fiori UIs or EML APIs.

> Similar validations were already handled in RAP100. Therefore, you will simply adopt the provided source code. We will use the validations for playing around with features such as **determine actions** and **side effects**.

Exercises:

- 6.1 - Implement the Validations of the *Travel* BO Entity
- 6.2 - Implement the Validations of the *Booking* BO Entity
- 6.3 - Preview and Test the enhanced Travel App
- Summary
- Appendix

> **Reminder:** Do not forget to replace the suffix placeholder `###` with your choosen or assigned group ID in the exercise steps below.

About Validations

▸ Click to expand!

Exercise 6.1: Implement the Validations of the *Travel* BO Entity

^Top of page

> Enhance the behavior implementation class of the *travel* entity with the business logic for the validation methods `validateCustomer`, `validateAgency`, and `validateDates`.

> There are two (2) ways to complete exercise 6.1:
>
> - Option 1: **This is the recommended option.** Replace the whole content of your behavior implementation class of the *travel* entity ZRAP110_BP_TRAVELTP_### with the provided source code document linked below and replace the placeholder `###` with your group ID. Save and activate the changes, then **procedd directly with Exercise 6.2.**
>
> Source code document: Behavior Implementation Class ZRAP110_BP_TRAVELTP_###
>
> - Option 2: Carry out the steps described below (6.1) in sequence.

▾ Click to expand!

Exercise 6.1.1: Implement the Validation `validateCustomer` of the *Travel* BO Entity

> Implement the validation `validateCustomer` which checks if the customer ID (`CustomerID`) that is entered by the consumer is valid. An appropriate message should be raised and displayed on the UI for each invalid value.

▾ Click to expand!

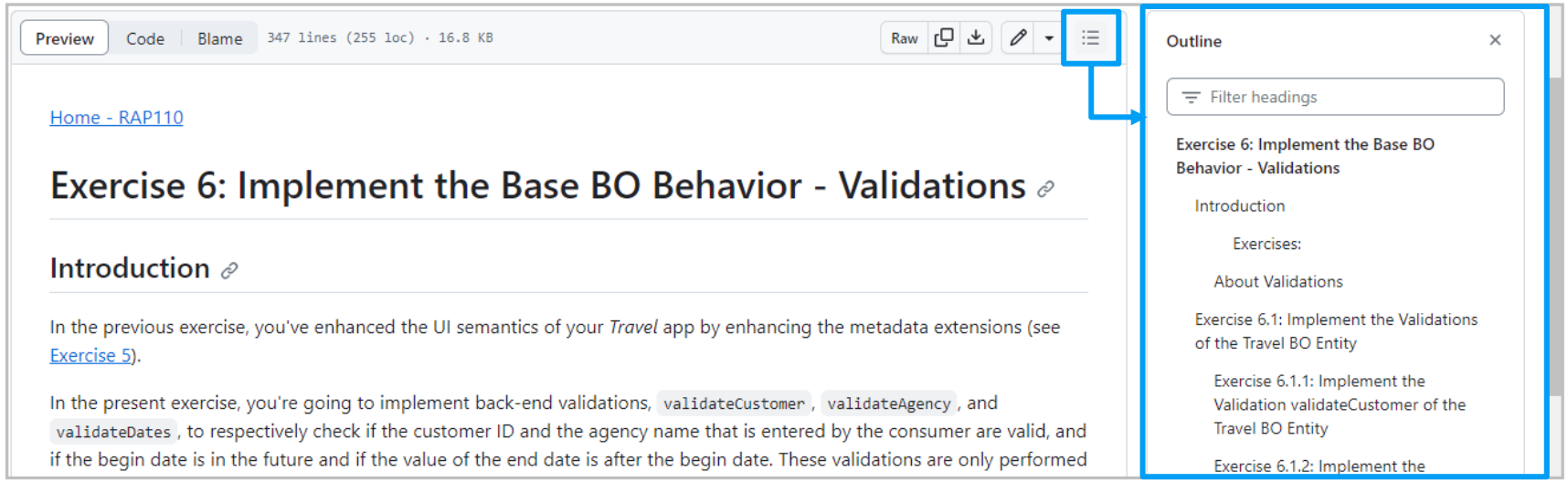
1. In your implementation class of the *travel* entity `ZRAP110_BP_TRAVELTP_###`, replace the current method implementation of `validateCustomer` with following code snippet.

   Replace all occurrences of the placeholder `###` with your group ID.

```
METHOD validateCustomer.
 "read relevant travel instance data
  READ ENTITIES OF ZRAP110_R_TravelTP_### IN LOCAL MODE
  ENTITY Travel
   FIELDS ( CustomerID )
   WITH CORRESPONDING #( keys )
  RESULT DATA(travels).
```

The AD181v hands-on exercises – Working on GitHub (2)

Use the Outline view to get a quick overview of the exercises.

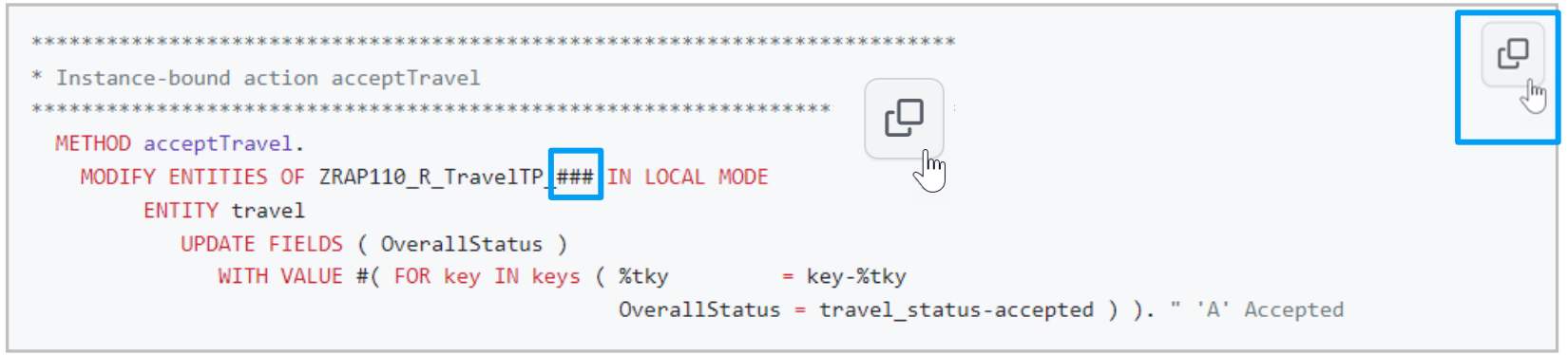


Use provided booster exercises to speed up the steps of familiar exercises

- 3.1 - How to handle this exercise 💡
- 3.2 - Define the Late Numbering and the Static Field Control
- 3.3 - Define the Validations
- 3.4 - Define the Actions
- 3.5 - Define the Determinations

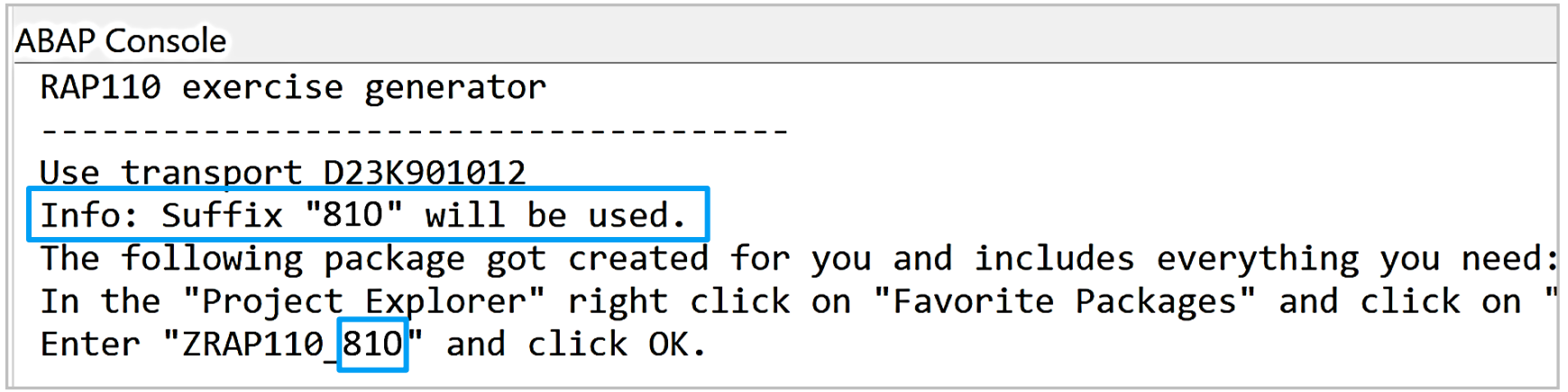
The AD181v hands-on exercises – Working on GitHub (3)

Make use of Copy raw contents button to easily copy code snippets



```
**********************************************************************
* Instance-bound action acceptTravel
**********************************************************************
  METHOD acceptTravel.
    MODIFY ENTITIES OF ZRAP110_R_TravelTP_### IN LOCAL MODE
         ENTITY travel
            UPDATE FIELDS ( OverallStatus )
               WITH VALUE #( FOR key IN keys ( %tky         = key-%tky
                                               OverallStatus = travel_status-accepted ) ). " 'A' Accepted
```

Always replace all occurrences of ### with your assigned suffix in code snippets

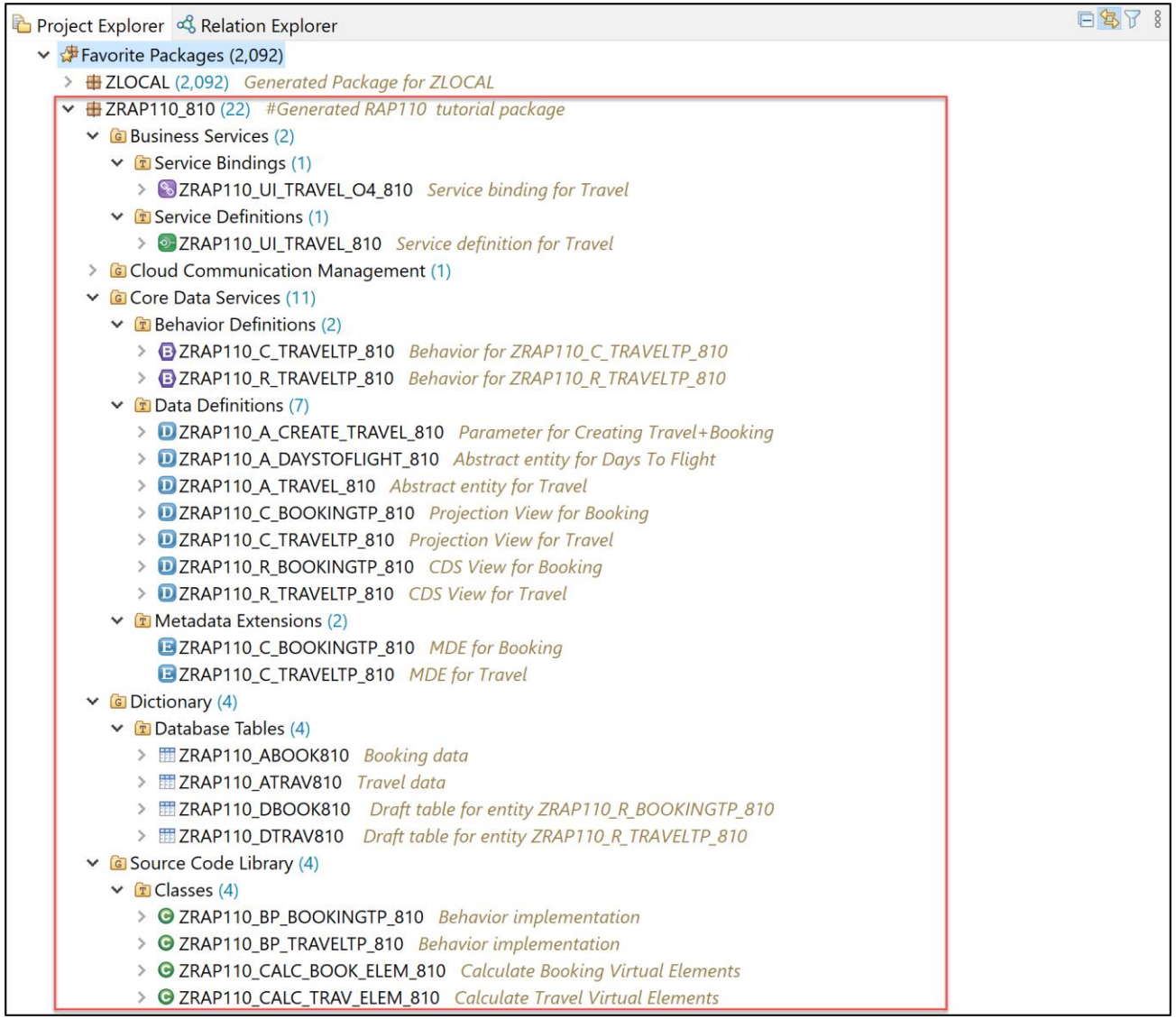


ABAP Console

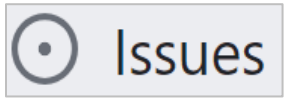
```
RAP110 exercise generator
-------------------------------------
Use transport D23K901012
Info: Suffix "810" will be used.
The following package got created for you and includes everything you need:
In the "Project Explorer" right click on "Favorite Packages" and click on "
Enter "ZRAP110_810" and click OK.
```

The AD181v hands-on exercises – Working on GitHub (3)

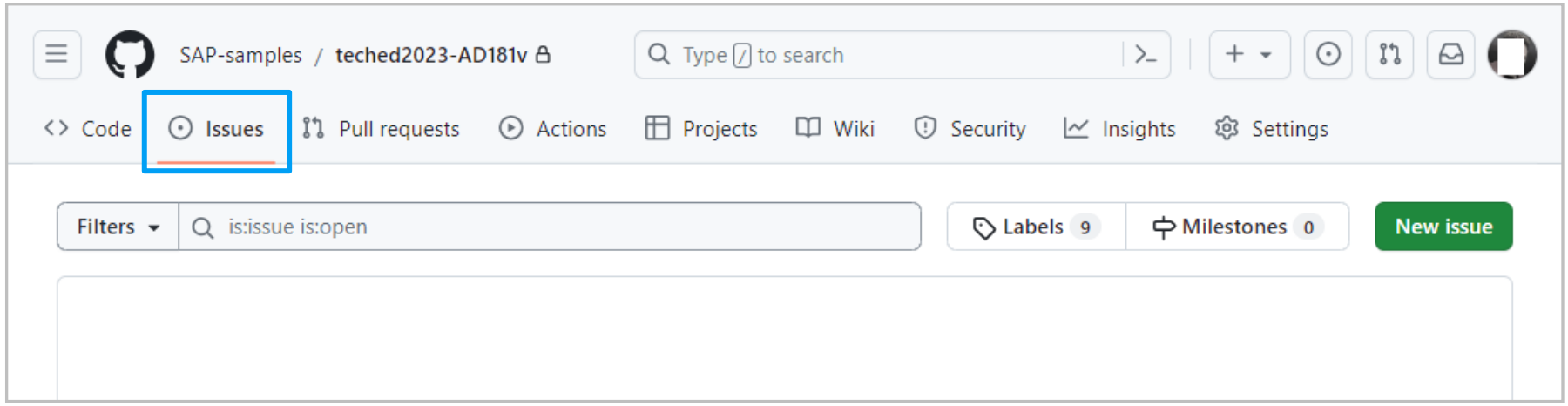
Your **generated exercise package** will look like this →



The AD181v hands-on exercises – Working on GitHub (4)



Open GitHub Issues to get help from the SAP experts.



COMPACT GLOSSARY

SUMMARY

Key takeaways

ABAP Cloud is the development model provided to build cloud-ready enterprise services, apps, and extensions on SAP BTP ABAP environment and all editions[1] of SAP S/4HANA, in the cloud and on-premises.

The **ABAP RESTful application programming model** (RAP) is at the **heart of ABAP Cloud** for efficiently building **transactional** OData-based services and SAP Fiori apps with built-in cloud qualities.

RAP best support **SAP HANA** and **SAP Fiori elements**.

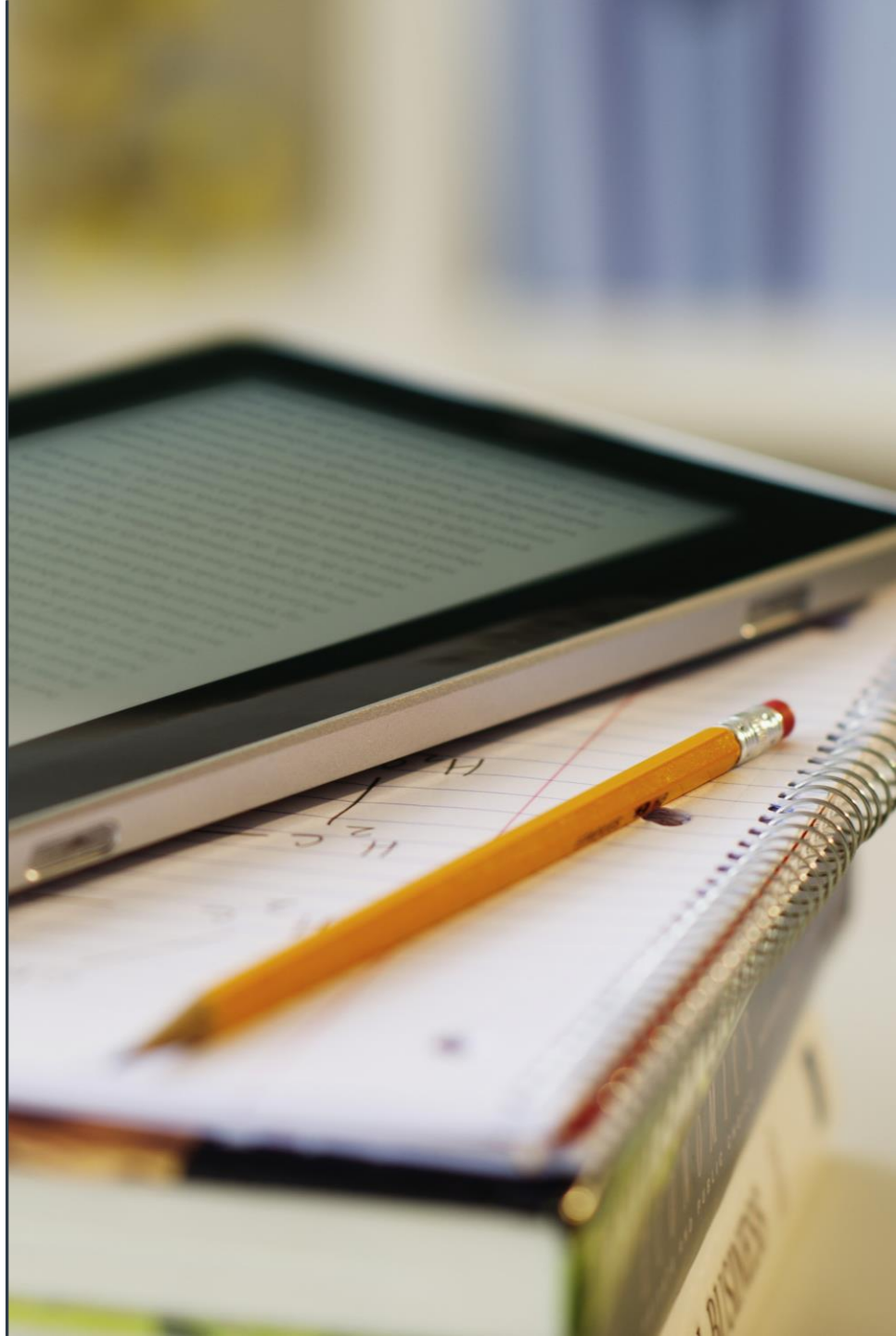
RAP is available on **SAP BTP ABAP Environment**, **SAP S/4HANA Cloud**, and **SAP S/4HANA** as of edition 1909.

The **RAP feature set** is enhanced **quarterly** in SAP BTP ABAP environment, **twice a year** in SAP S/4HANA Cloud, and **every two years** in SAP S/4HANA and SAP S/4HANA Cloud, private edition.

What's New: SAP BTP ABAP environment | SAP S/4HANA | SAP S/4HANA Cloud

What's Next: ABAP Platform Roadmap Information for all SAP products

[1] SAP S/4HANA any premise or SAP S/4HANA Cloud, private edition release ≥ 2022
SAP S/4HANA Cloud, public edition release ≥ 2208, 3-system landscape required



Learn more about ABAP Cloud at SAP TechEd 2023

Check the virtual Session Catalog: SAP TechEd Virtual in 2023

Check out these insightful virtual sessions to learn more about accelerating development with pro-code tools: AD103v, AD107v, DT103v, DT200v, DT182v, DT187v

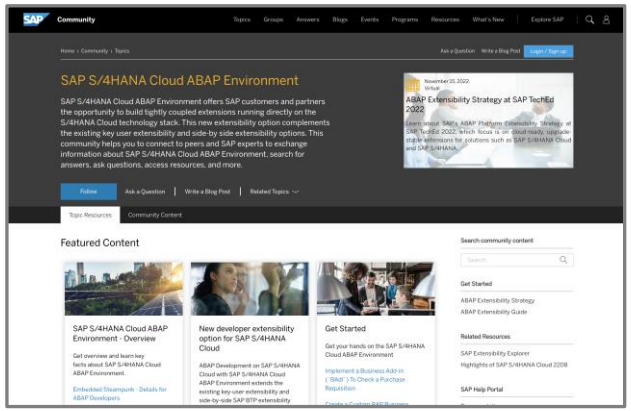
See blog post ABAP Cloud at SAP TechEd 2023

NEW!

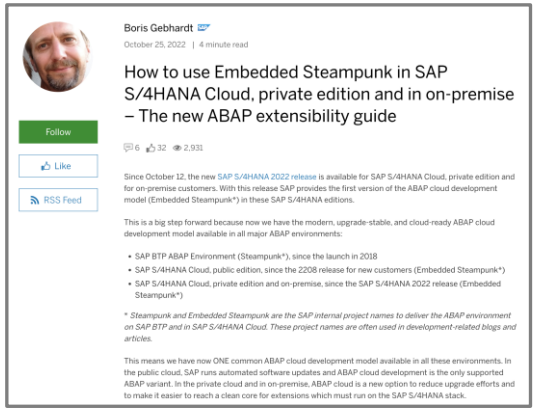
SAP BTP Developer's Guide

https://url.sap/kr99u2

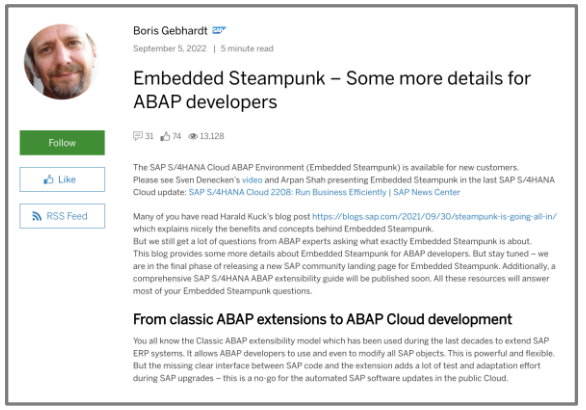
More information on ABAP Cloud



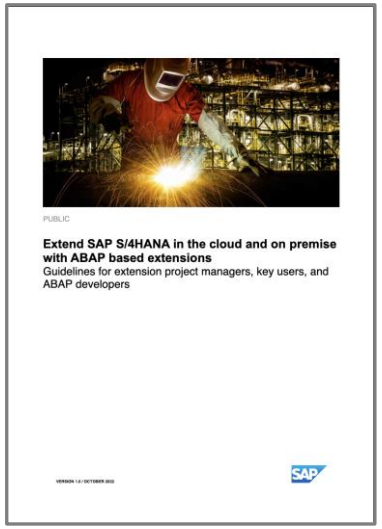
SAP Community



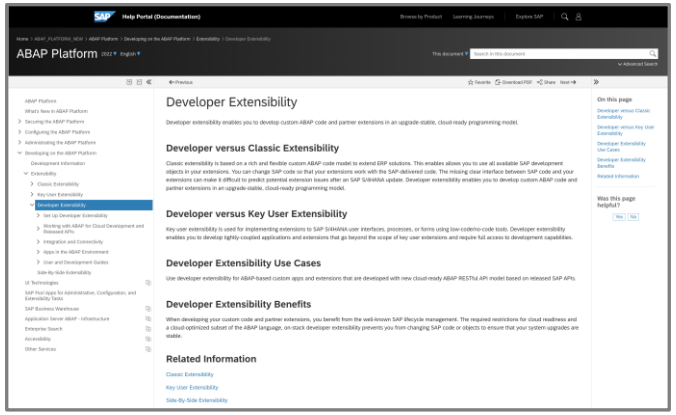
How to use Embedded Steampunk in SAP S/4HANA Cloud, private edition and in on-premise – The new ABAP extensibility guide | SAP Blogs



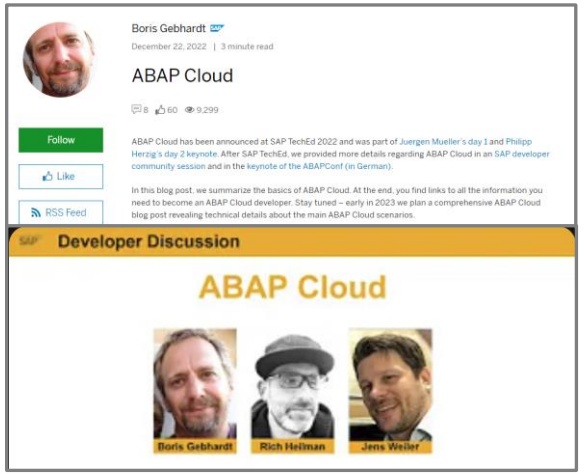
Embedded Steampunk – Some more details for ABAP developers | SAP Blogs



The new ABAP extensibility guide



SAP documentation



ABAP Cloud | SAP Blogs

Developer Discussion: ABAP Cloud

More information on RAP

State-of-the-Art ABAP Development with the ABAP RESTful Application Programming Model (RAP) | SAP Community

Modernization with RAP | SAP Blogs

Acquire Core ABAP Skills | Practicing Clean Core Extensibility for SAP S/4HANA Cloud | SAP Learning Journey

ABAP Cloud at SAP's Devtoberfest in 2023 | ABAP Cloud sessions at SAP TechEd 2023

SAP Fiori elements Feature Showcase App with RAP

What's New in RAP: SAP BTP ABAP Environment | SAP S/4HANA | SAP S/4HANA Cloud

Outlook: ABAP Platform Roadmap Information

Public SAP Web sites

ABAP Development Community: www.sap.com/community/topic/abap.html
SAP BTP ABAP Environment Community: https://community.sap.com/topics/btp-abap-environment
SAP S/4HANA Cloud ABAP Environment Community: https://community.sap.com/topics/s4hana-cloud-abap-environment
ABAP Testing and Analysis Community: https://community.sap.com/topics/abap-testing-analysis
SAP products: www.sap.com/products

SAP training and certification opportunities

www.sap.com/education – e.g. trainings S4D437, S4D430, HA400, and S4D400
learning.sap.com/learning-journey – e.g., search for ABAP or ABAP Cloud

Make your career growth real

Upskill and engage on Application Development and Automation

Free learning
Upskill and prepare for certification

25% discount
On SAP Certification exams

Expand your network
Engage with experts and share knowledge

GET REAL BENEFITS: learning.sap.com/teched

37% Of IT certification candidates received salary increases after earning a certification*

92% Are more confident in their abilities*

*"2023 Value of IT Certification Candidate Report," Pearson VUE, 2023.

Thank you

Contact information:

Carine Tchoutouo Djomo
Product Manager for ABAP Platform

Merve Temel
Product Manager for ABAP Platform

APPENDIX

COMPACT GLOSSARY

About some of the used RAP features – Compact Glossary

→ BACK TO MAIN PRESENTATION

UI Semantics with CDS Annotations

Various UI-specific CDS annotations can be used to solve the most common UI layout tasks in SAP Fiori elements apps built with RAP. UI annotations represent semantic views on business data through the use of specific patterns that are independent of UI technologies. For more information, see Develop UI Specifics | CDS Annotations | SAP Fiori elements Showcase App for RAP And ABAP CDS.

Support for Large Objects (aka OData Streams)

You can enable your RAP application for maintaining large objects (LOBs). By doing so, you provide end users the option to incorporate external binary files or text files when editing entity instances. For more information, see Working with Large Objects

Virtual Elements in CDS Projection Views

Virtual elements represent transient fields in business applications. They are used to define additional CDS elements that are not persisted on the database, but calculated during runtime using ABAP classes that implement the virtual element interface. They are defined at the level of CDS projection views as additional elements within the SELECT list. For more information, see Virtual Elements.

About some of the used RAP features – Compact Glossary (2)

→ BACK TO MAIN PRESENTATION

Late Numbering

Numbering is the process of setting values for primary key fields of entity instances during runtime. Early and late numbering are supported in RAP. In a **late numbering** scenario, the key values are always assigned internally without consumer interaction after the point of no return in the interaction phase has passed, and the SAVE sequence is triggered. For more information, see Numbering.

Validations

Validations are used to ensure the data consistency. Back-end are defined in the BO behavior definitions and implemented in the respective behavior implementation classes. Front-end validations are used to improve the user experience by providing faster feedback and avoiding unnecessary roundtrips. They can be defined in the BO data model using value helps. For more information, see Validations.

Determinations

A determination is an optional part of the business object behavior that modifies instances of business objects based on trigger conditions. A determination is implicitly invoked by the RAP framework if the trigger condition of the determination is fulfilled. Trigger conditions can be modify operations and modified fields. A determination can be triggered on modify or on save. For more information, see Determinations.

About some of the used RAP features – Compact Glossary (3)

→ BACK TO MAIN PRESENTATION

Actions

Actions are specified in behavior definitions and implemented in ABAP behavior pools. By default, actions are related to instances of a BO entity. The addition static allows you to define a static actions that are not bound to any instance but relates to the complete entity. The addition internal define a private action that can only be called within the given BO. For more information, see Actions.

Functions

A function in RAP is a custom read-operation that is part of the business logic.
Functions perform calculations or reads on business objects without causing any side effects. Functions don't issue any locks on database tables and you can't modify or persist any data computed in a function implementation. For more information, see Functions.

RAP Business Events

Business events provide the opportunity of light-weight, decoupled process integration based on standardized and stable APIs and they are now a native part of RAP. With the RAP Business Event Bindings Editor, you can create RAP Event Bindings which are needed to provide a mapping between the definition of RAP Events via behavior definition (BDEF) and the external representation of Business Events. For more information, see RAP Business Events.

Business events can be consumed locally using an event handler class or remotely using the SAP Event Mesh.

About some of the used RAP features – Compact Glossary (4)

→ BACK TO MAIN PRESENTATION

Static and Dynamic Feature Control

Feature control offers developers the possibility to determine, which entities of your business object should be create-, delete- and update-enabled, so that they can be modified during consumption using EML or OData services. It also allows developers to control which (UI) fields of an entity are read-only or which actions in which usage scenarios are enabled or disabled for execution by the end users.

The availability of feature control values is modeled in a behavior definition. Unlike static feature control, instance feature control requires not only a definition but also an implementation in a handler class of the behavior pool. Therefore, we also talk about dynamic feature control in case of instance feature control.

Feature control can be related to an entire entity or to individual elements of an entity, such as individual fields or operations. For more information, see Feature Control.