

1 Goal of this tutorial

Set up the SAP WebIDE Eclipse based development environment and use it with Chrome browser (<http://localhost:8080/webide/index.html>). Create a simple Hello World Application, upload it to the ABAP server and test it.

2 General

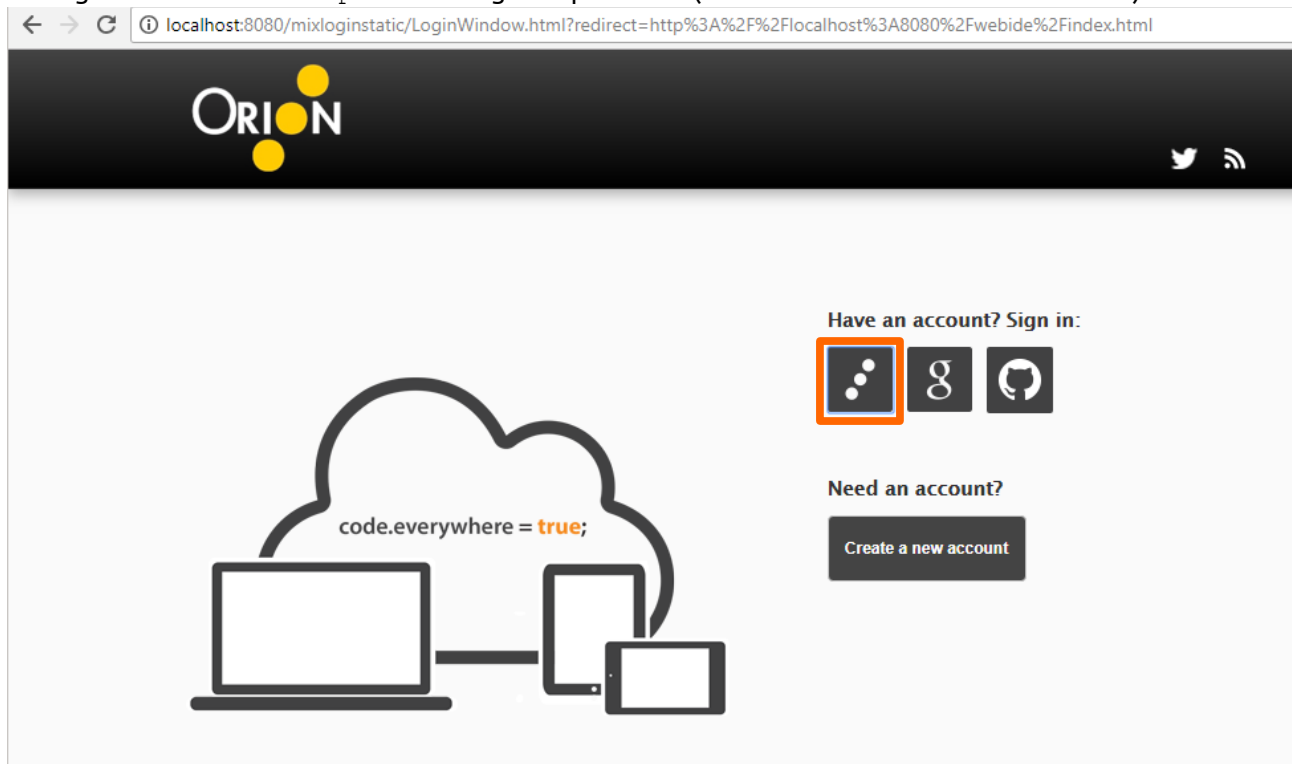
In the following examples replace XXX by your id and Vorname / Nachname by your name – you can use all materials, internet, work in teams of two, but you have to create your own results.

Collect the artefacts of your result as requested below in a result document

SAPUI5_01_<name>_<vorname>.docx [Total: 19P].

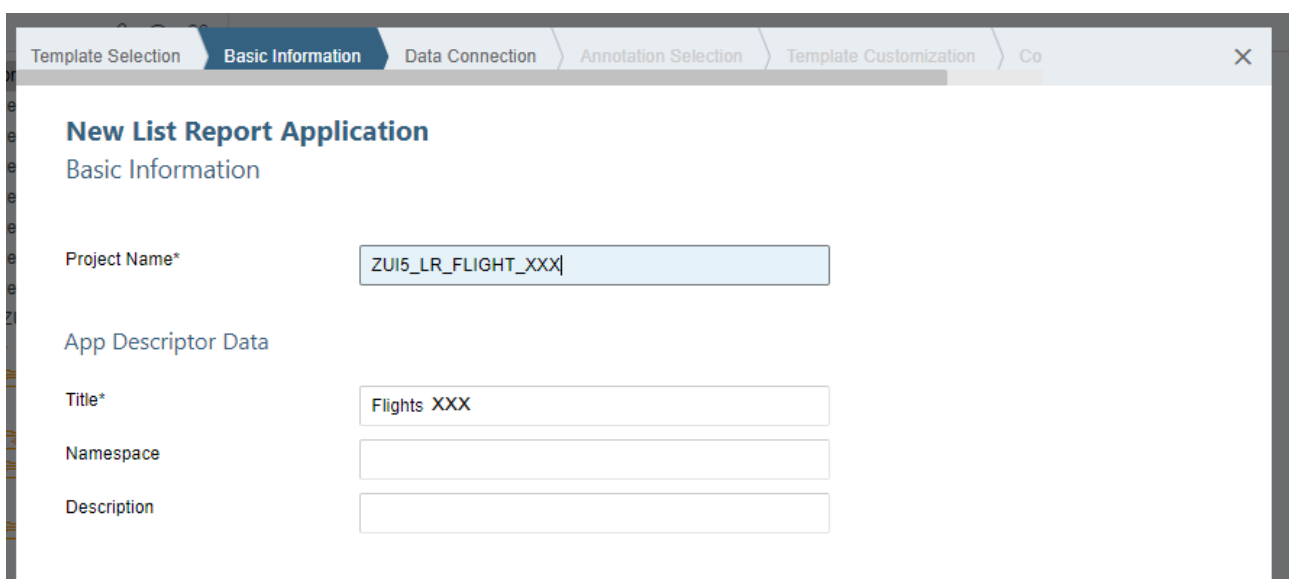
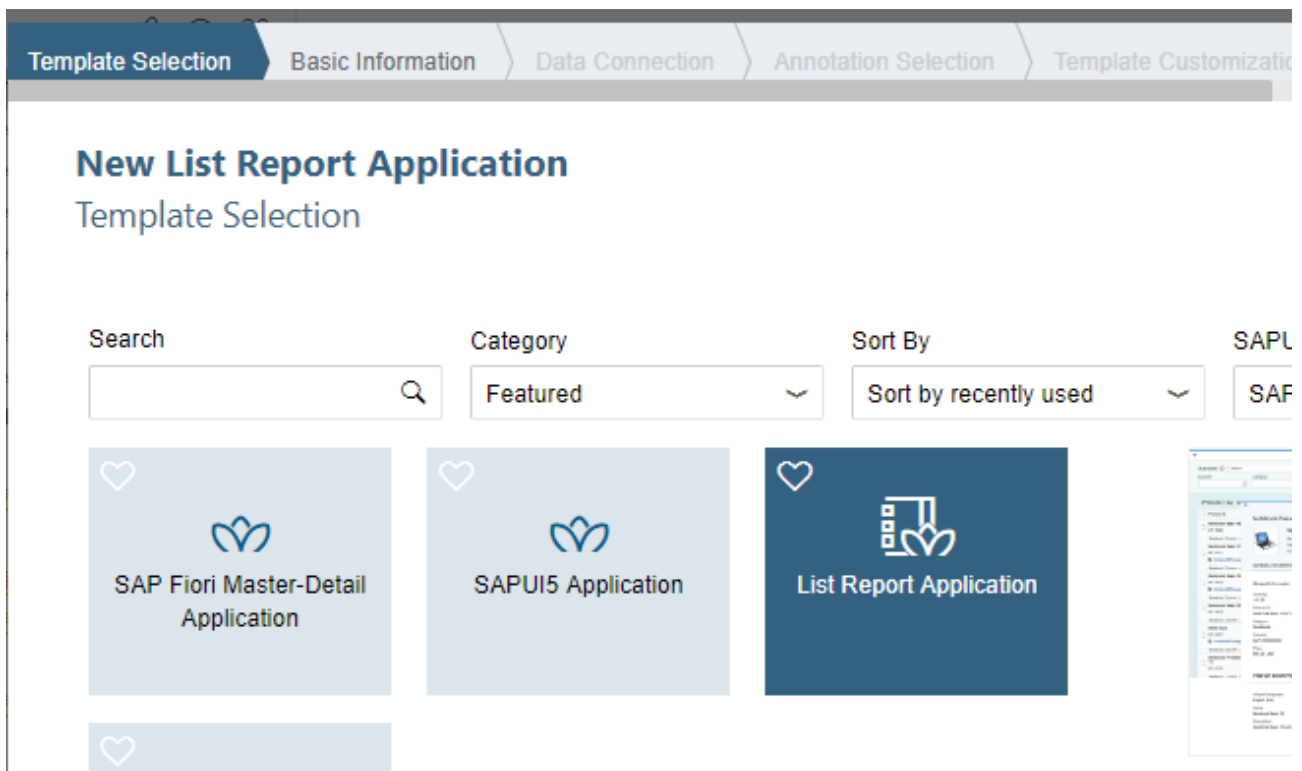
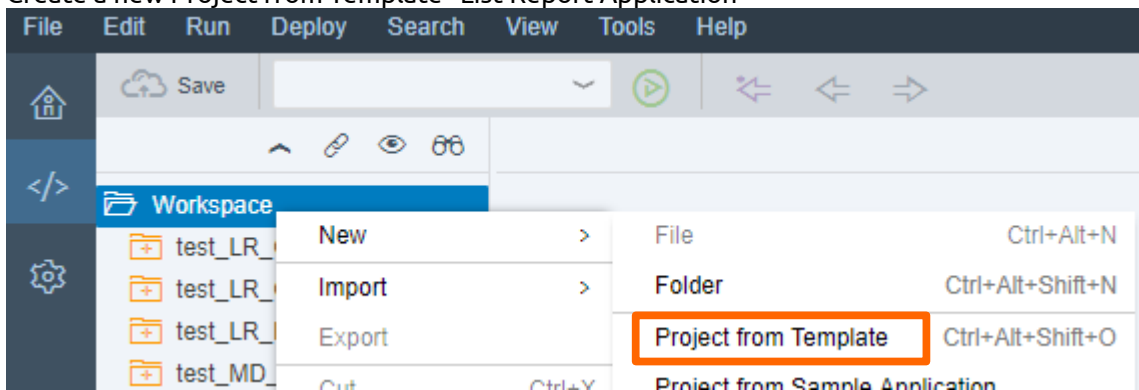
3 Setup of IDE

- 1 Extract the `sapWebIDE-PE.zip` (on class share) inside your VM on your Desktop as directory `sapWebIDE-PE`
- 2 Start `sapWebIDE-PE/orion.exe` which will open a command shell
- 3 Open the link “Chrome – SAPWebIDE..” in Chrome
- 4 Log in with user `developer` and the gibbix password (click on the icon with the three dots):



5 First: Generating List Report Application using an OData-Service

Create a new Project from Template “List Report Application”



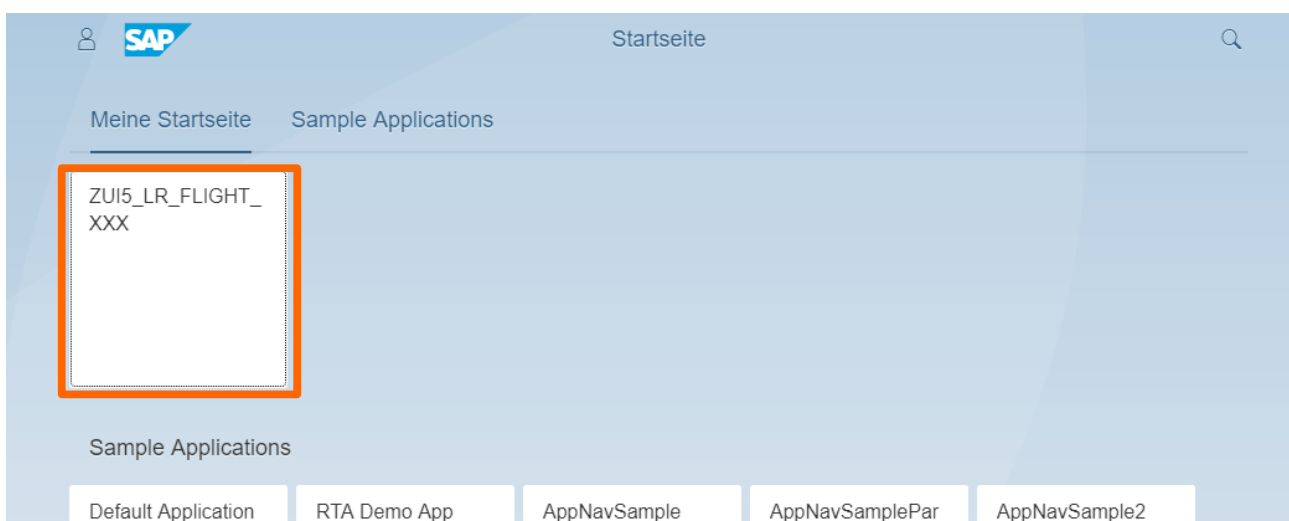
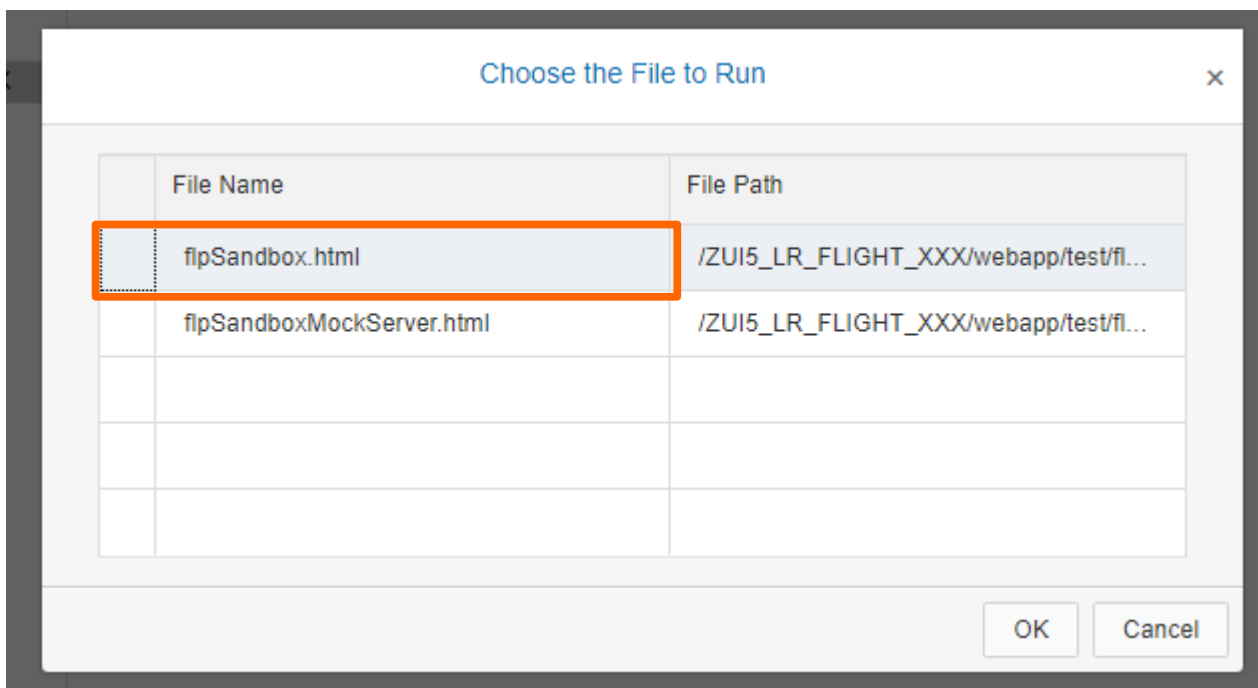
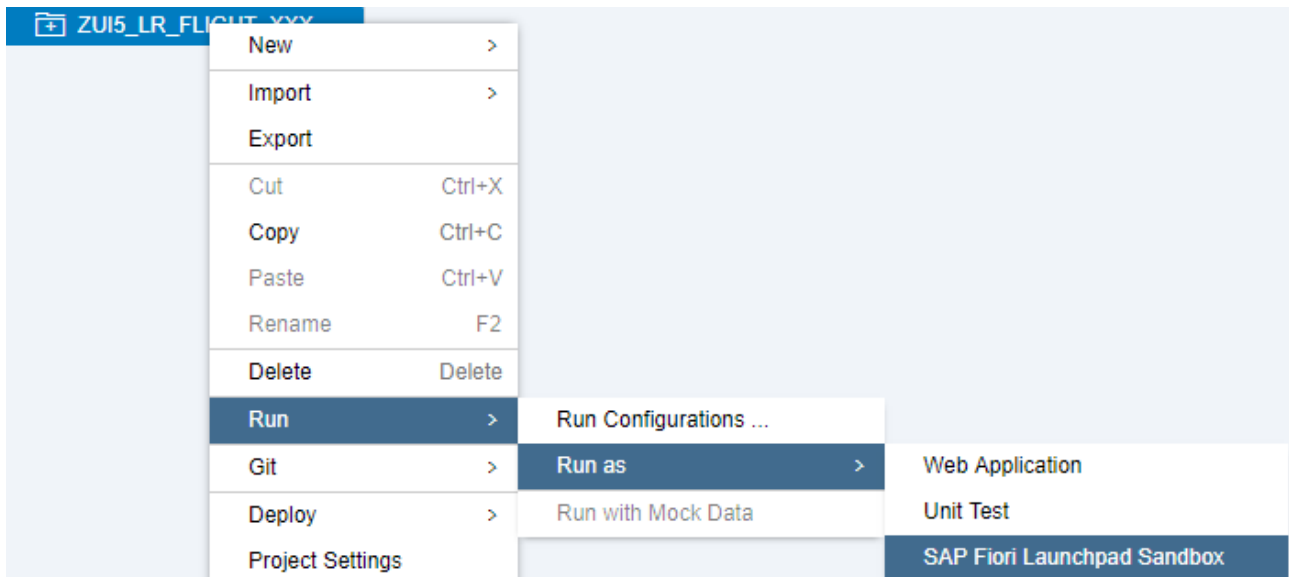
Select your SAP System and there the Service RMTSAMPLEFLIGHT

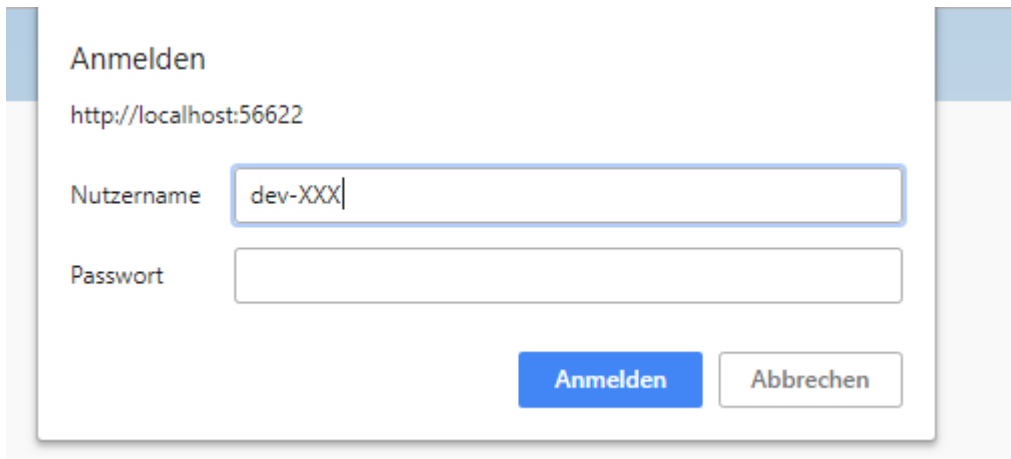
The screenshot shows the 'Data Connection' step of the 'New List Report Application' wizard. The 'Service Catalog' is selected in the left sidebar. The system 'G12' is chosen in the dropdown. A search bar is present with the text 'Ser...' and a 'Show I' button. A table lists available services:

Service	Descri...
> (→) LORD_MY_QUOTATION_SF	Meine And...
> (→) LORD_ODATA_ORDER_SR	OData-Ser...
> (→) ME2STAR_SRV	Gatewav...
> (→) NOTIFICATIONSTORE	NOTIFICA...
> (→) PP_MRP_COCKPIT_SRV	MRP-Cockpit
▼ (→) RMTSAMPLEFLIGHT	RMTSAM...
> [Grid Icon] BookingCollection	
> [Grid Icon] CarrierCollection	

The screenshot shows the 'Template Customization' step of the 'New List Report Application' wizard. The 'Data Binding' section has two dropdowns: 'OData Collection...' set to 'FlightCollection' and 'OData Navigatio...' set to 'flightBookings'. A preview window on the right shows a sample application layout with a list of flights and a detailed view of a flight's bookings.

..for showing the flights in the first list and the bookings for each flight in a second list.





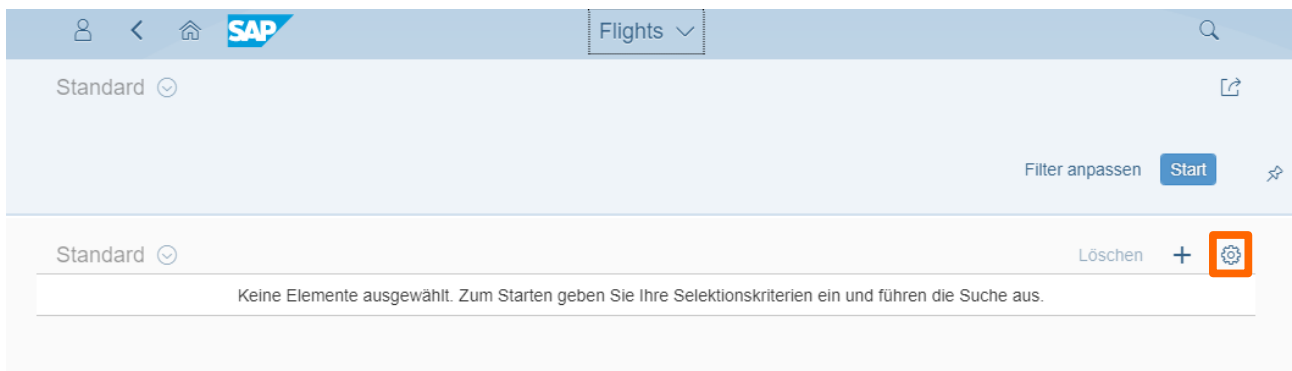
Anmelden

http://localhost:56622

Nutzername

Passwort


Anmelden Abbrechen



SAP Flights

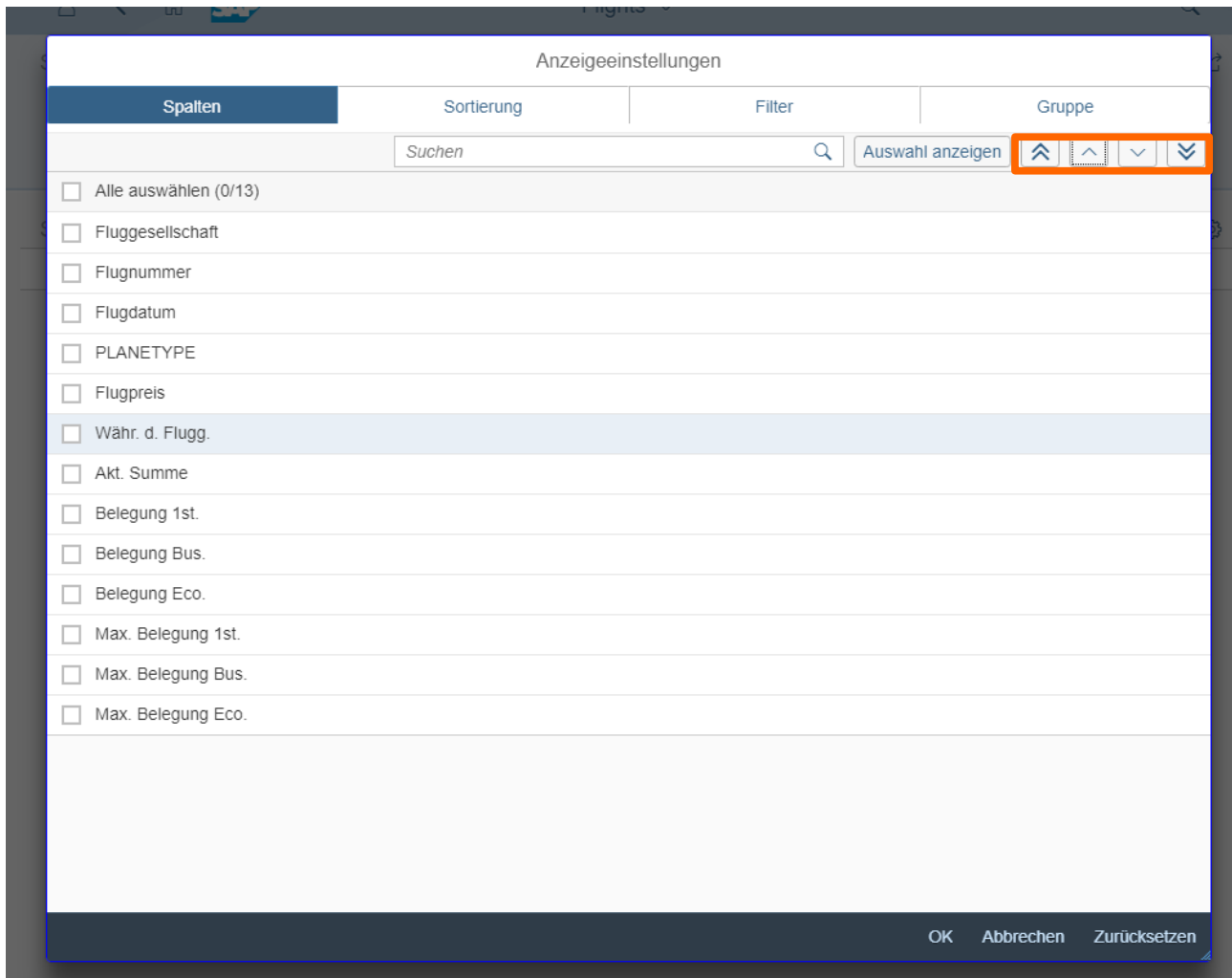
Standard

Filter anpassen **Start**

Standard Löschen + 

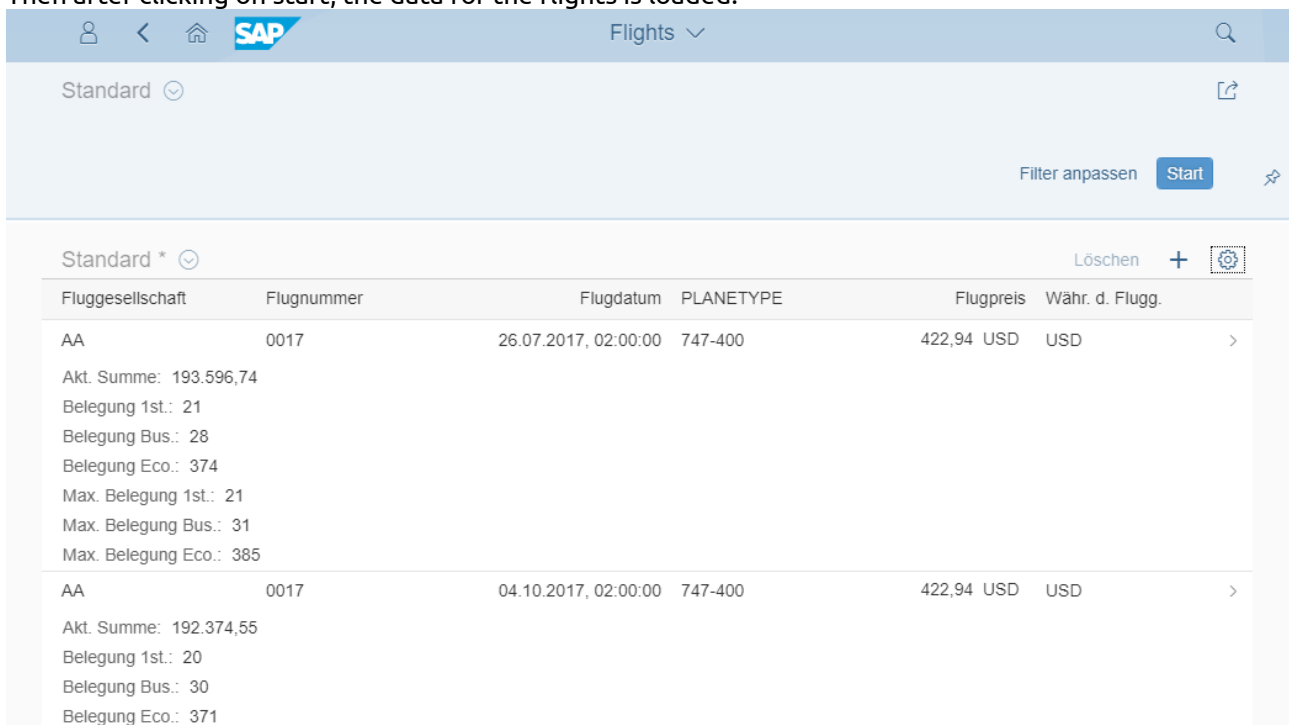
Keine Elemente ausgewählt. Zum Starten geben Sie Ihre Selektionskriterien ein und führen die Suche aus.

Select the wheel-icon to choose the data-columns/attributes to show.



You can also arrange the sequence of the attributes

Then after clicking on start, the data for the flights is loaded:



SAP UI5 – Hello World

On the second screen you can also select the attributes to show.
After selecting them, navigate back and forth to load the data for the bookings

The screenshot shows the SAP UI5 'Flights' application. The top navigation bar includes a user icon, back and home arrows, the SAP logo, the title 'Flights' with a dropdown arrow, and a search icon. On the right side of the bar are buttons for 'Bearbeiten', 'Löschen', and a share icon. Below the bar, there are two tabs: 'General Information' (selected) and 'Second Facet'. The 'General Information' section displays a list of booking details for a specific flight. The 'Second Facet' section contains a table with flight data.

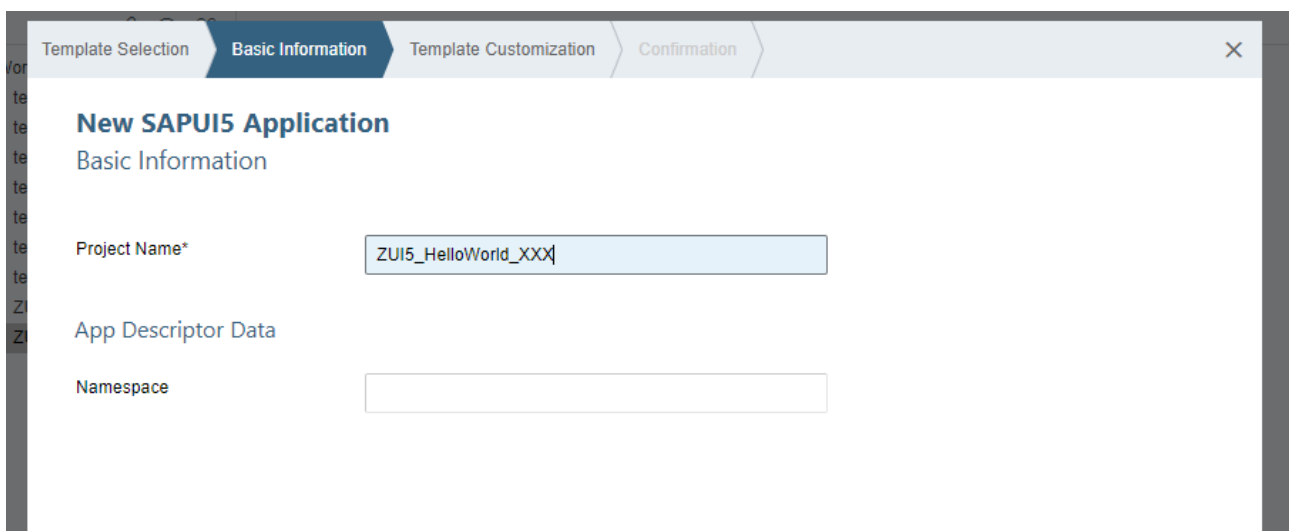
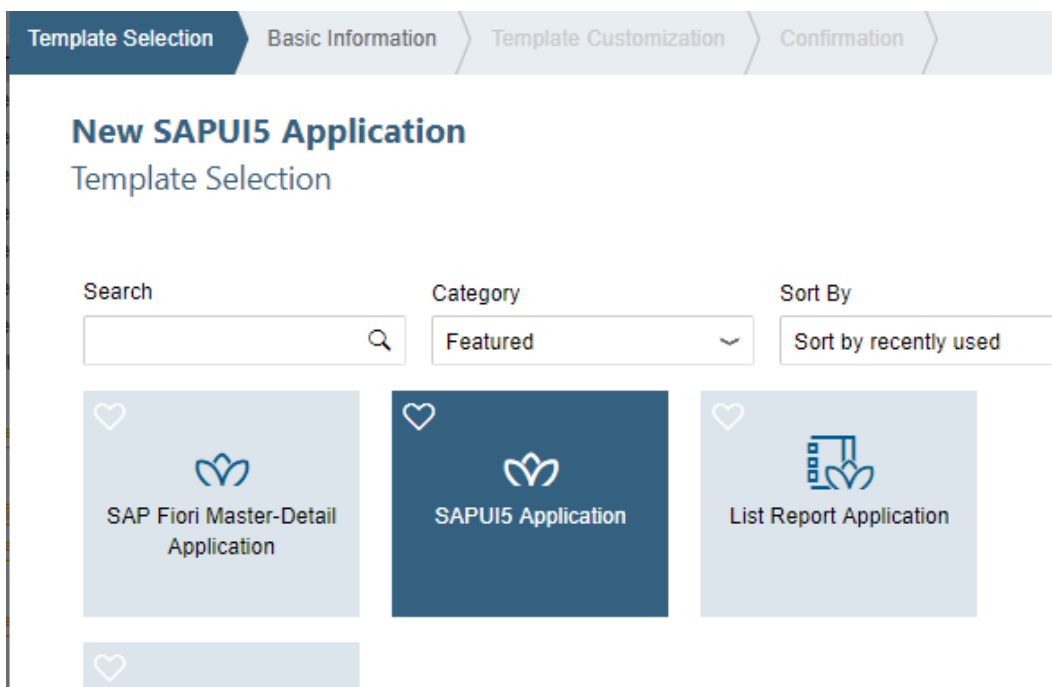
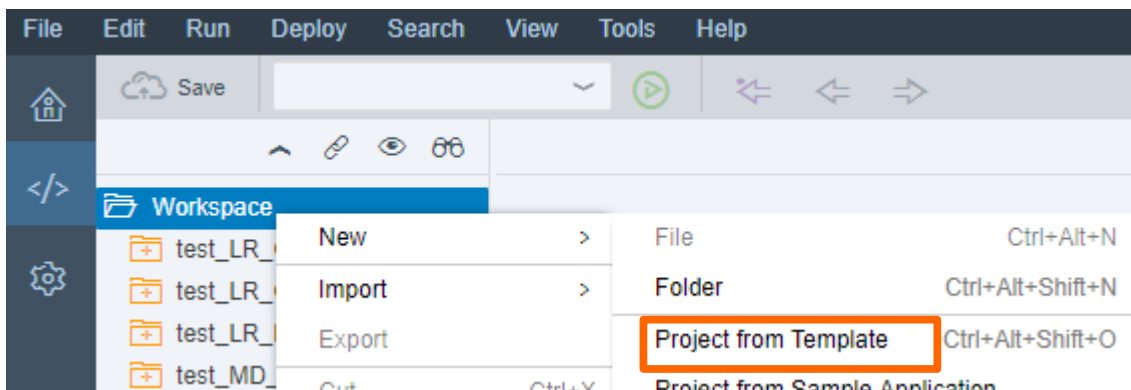
Kundennummer	Anrede	Passagiername	Fluggesellschaft	Flugnummer	Flugdatum
00002619		Anne-Marie Cesari-Tran	AA	0017	26.07.2017, 02:00:00 >

Below the table, a detailed view of the selected booking is shown:

- Buchungsnummer: 00000001
- Buchungsdatum: 25.06.2017, 02:00:00
- Reisebüro nr.: 00000325
- Betrag: 922,01 EUR
- Betrag: 845,88 USD
- G/P-Kunde: P
- Gepäckgew.: 19,4000 KGM
- Klasse: C
- Maßeinheit: KGM
- Raucher:
- Rechnungsst.: X
- reserviert:
- Stornokennzeichen:
- Verkaufsstelle: 00000000
- Währ. d. Flugg.: USD
- Zahl.Währung: EUR
- Pass. Gebdatum: 27.05.2018, 02:00:00

→ Create a screenshot of the flights view and the booking view and collect it in the result document [4P]

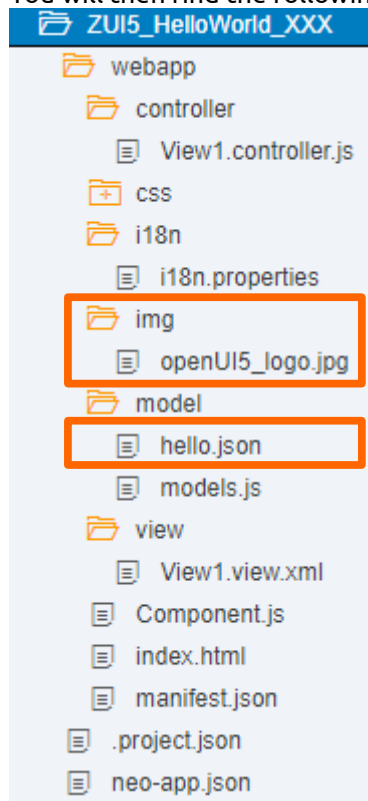
6 Second: Create a UI5 Application “HelloWorld”



The screenshot shows the 'Template Customization' step of the 'New SAPUI5 Application' wizard. The wizard has four steps: Template Selection, Basic Information, Template Customization (current), and Confirmation. The main heading is 'New SAPUI5 Application' followed by 'Template Customization'. There are two input fields: 'View Type*' with a dropdown menu showing 'XML' and a downward arrow, and 'View Name' with a text box containing 'View1'.

Choose XML for the view type.

You will then find the following structure:



The orange files need to be created as follows.

SAP UI5 – Hello World

1. Create folder `img` and import logo-image (from WebIDE-Directory)
2. Create `hello.json` in model folder, using your name

```
hello.json x
1 {
2   "user" : {
3     "name" : "Vorname Nachname"
4   }
5 }
```

3. To use the `hello.json` as model, define the usage in `manifest.json` (blue)

```
manifest.json x
67 },
68 "models": {
69   "i18n": {
70     "type": "sap.ui.model.resource.ResourceModel",
71     "settings": {
72       "bundleName": "ZUI5_HelloWorld_MSC.i18n.i18n"
73     }
74   },
75   "hello": {
76     "type": "sap.ui.model.json.JSONModel",
77     "uri": "model/hello.json"
78   }
79 },
80 "resources": {
```

4. Add translatable texts in `i18n.properties`

```
i18n.properties x
1 title=My Title
2 appTitle = App Title
3 appDescription=App Description
4
5 /* hello example */
6 helloButtonText=Say Hello!
7 helloMsg=Hello {0}
```

Now we are prepared for the next steps.

SAP UI5 – Hello World

First we will only use the `index.html`. First comment out the part called "third:" and add the blue code.

```
20 <script>
21     sap.ui.getCore().attachInit(function() {
22
23         /* third: comment first and second */
24         new sap.m.Shell({
25             app: new sap.ui.core.ComponentContainer({
26                 height : "100%",
27                 name : "ZUI5_HelloWorld_XXX"
28             })
29         }).placeAt("content");
30
31         /* second: display an image with click-handler */
32         var oImage = new sap.m.Image({
33             src: "img/openUI5_logo.jpg",
34             decorative: false,
35             alt: "SAPUI5 Logo",
36             press: function(){
37                 alert("Hello User XXX :");
38             }
39         }).placeAt("content");
40
41         /* first: write some text */
42         var oTxt = new sap.m.Text();
43         oTxt.setText("Hello UI5 World \n on a new line");
44         oTxt.placeAt("content");
45
46     });
47 </script>
48 </head>
```

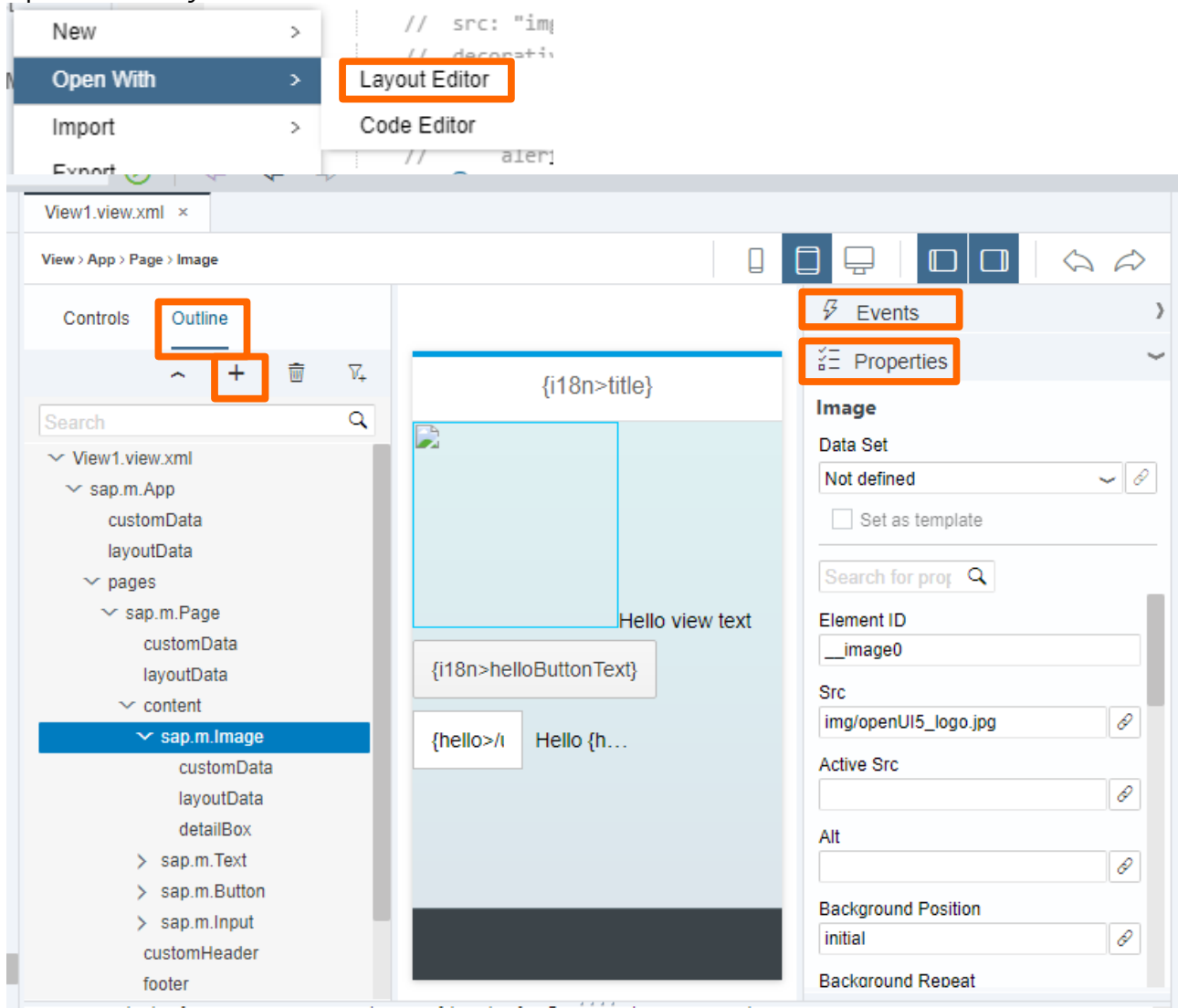
Test it by running it as web application. It should show the text and the image. Clicking on the image, it should show the message as popup.

→ Create a screenshot after executing and collect it in the result document [3P]

SAP UI5 – Hello World

Then comment out the blue code and activate the part called “third:”. Now we continue preparing the view.

Open it in the Layout editor:



Add an image control and a text control. The code marked blue below will be added in the view.

Reference: <https://sapui5.hana.ondemand.com/#/api>

Add a press event to the image element and create a new function `onShowHelloView` for that. Implement the function that has been added in the controller (marked blue below) and remember to add the `MessageToast` module in line 3 and 4 of the controller.

Test it by running it as web application. It should show the text and the image. Clicking on the image, it should show the message as popup and then as `MessageToast`.

→ Create a screenshot after executing and collect it in the result document [6P]

```

View1.view.xml ×
1 <mvc:View xmlns:html="http://www.w3.org/1999/xhtml" xmlns:mvc="sap.ui.core.mvc" xmlns="sap.m" controllerName=
2 <App>
3 <pages>
4 <Page title="{i18n>title}">
5 <content>
6
7 <Image width="140px" height="140px" id="__image0" src="img/openUI5_logo.jpg" press="onShowHel
8 <Text text="Hello view text" id="__text5"/>
9
10 <Button
11     text="{i18n>helloButtonText}"
12     press="onShowHelloButton"/>
13 <Input
14     value="{hello>/user/name}"
15     description="Hello {hello>/user/name}"
16     valueLiveUpdate="true"
17     width="60%"/>
18
19 </content>
20 </Page>
21 </pages>
22 </App>
23 </mvc:View>

```

```

*View1.controller.js ×
1 sap.ui.define([
2     "sap/ui/core/mvc/Controller",
3     "sap/m/MessageToast" /* loading MessageToast module*/
4 ], function(Controller, MessageToast) {
5     "use strict";
6     return Controller.extend("ZUI5_HelloWorld_MSC.controller.View1", {
7         /**
8          * @memberOf ZUI5_HelloWorld_MSC.controller.View1
9          */
10        onShowHelloView: function() {
11            //This code was generated by the layout editor.
12            alert("Hello XXX from view controller");
13            MessageToast.show("Hello nice World");
14        },
15
16        onShowHelloButton : function () {
17            // read msg from i18n model
18            var oBundle = this.getView().getModel("i18n").getResourceBundle();
19            var sHello = this.getView().getModel("hello").getProperty("/user/name");
20            var sMsg = oBundle.getText("helloMsg", [sHello]);
21            // show message
22            MessageToast.show(sMsg);
23        }
24    });
25 });

```

SAP UI5 – Hello World

Then add the code for the button and input element in the view and add the function `onShowHelloButton` in the controller (code shown above)

Test it by running it as web application. It should a new button, input field with the text from the json-model and its (dynamically updating) description text to the right.

Type “Bern” after the content in the input field.

Clicking on the button should show a greeting message as `MessageToast`.

→ *Create a screenshot after executing and collect it in the result document [6P]*