How to Extend a Fiori Application: Purchase Order Approval

Applicable Releases:
SAP Web IDE 1.4

Version 2.0 - October 2014
## Document History

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Authored By</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Customer Experience Group</td>
<td>First release of this guide</td>
</tr>
<tr>
<td>2.0</td>
<td>Customer Experience Group</td>
<td>Update for SAP Web IDE release 1.4</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS

1. Business Scenario ........................................................................................................... 4
2. Background Information .................................................................................................. 4
3. Prerequisites .................................................................................................................... 4
4. Step-by-Step Procedure .................................................................................................. 5
   4.1 Start creating a new EXTENSION Project ................................................................. 5
   4.2 Select Purchase Order Approval app from SAP Gateway ....................................... 6
   4.3 Extend the app by hiding an existing control ............................................................ 11
   4.4 Extend the app by adding a new UI field to an extension point ............................... 13
   4.5 Extend the app by replacing a view ............................................................................. 18
   4.6 Extend the existing OData service in Gateway Service Builder .............................. 23
   4.7 Replace the OData Service by an extended OData Service ..................................... 23
   4.8 Extend the data model using the EDMX editor, then add a reference to the new data property in the UI logic ................................................................. 28
   4.9 Deploy the application to SAP Gateway ..................................................................... 33
   4.10 Verify successful deployment of the app ................................................................. 36
1. BUSINESS SCENARIO
SAP Web IDE is a next-generation cloud-based meeting space where multiple project stakeholders can work together from a common web interface -- connecting to the same shared repository with virtually no setup required. It includes multiple interactive features that allow you to collaborate with your colleagues and accelerate the development of your HTML5/UI5 applications.

2. BACKGROUND INFORMATION
In this How-To Guide, we will show you how to extend a standard SAP Fiori application using a SAP Web IDE. The different possibilities of extending a Fiori app view are shown (hide a control, extend by implementing an extension point, replace a view, replace a service).

3. PREREQUISITES
To connect to your SAP Web IDE system, open the browser and enter the URL of your system.
4. **STEP-BY-STEP PROCEDURE**

The following major steps will be described in detail step-by-step with screenshots for every step.

1. Start creating a new extension project
2. Select Purchase Order Approval app from SAP Gateway (GM0)
3. Extend the app by hiding an existing control
4. Extend the app by adding a new UI field to an extension point
5. Extend the app by replacing a view
6. **OPTIONAL** - Go into the Gateway Service Builder and show how to extend the existing OData service *(separate document by Andre Fischer)*
7. **OPTIONAL** - Replace the OData Service by an extended OData Service (assuming the existing OData Service has been modified by adding a new field)
8. **OPTIONAL** - (if OData Service replacement is not shown) - For demo purposes, extend the data model using the EDMX editor, then add a reference to the new data property in the UI logic
9. Run the application with mock data
10. Deploy the application back to SAPUI5 ABAP Repository
11. Verify that the app is deployed by importing the app from SAP Gateway to SAP Web IDE

### 4.1 Start creating a new EXTENSION Project

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Start SAP Web IDE from your browser.</td>
<td><img src="image1.png" alt="Screenshot" /></td>
</tr>
<tr>
<td>2. Click on Local folder and start by creating a new extension project <em>(File &gt; New &gt; Extension Project)</em></td>
<td><img src="image2.png" alt="Screenshot" /></td>
</tr>
</tbody>
</table>
## 4.2 Select Purchase Order Approval app from SAP Gateway

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
</table>
| 1. From step 1 in the wizard, click **Remote...**, and select **SAPUI5 ABAP Repository** to access the remote SAP Gateway system which has the standard SAP Fiori PO Approval application deployed. | ![Screenshot 1](image1)

2. In the Select Application from **SAPUI5 ABAP Repository** dialog, select the SAP Gateway **System GM0** (GM0 Client 200). | ![Screenshot 2](image2)

3. All available applications on GM0 are shown. | ![Screenshot 3](image3)
4. Search for "purchase" to show search and filter capabilities

5. Select MM_PO_APV Approve Purchase Orders and click OK.
6. You may change the proposed project name to e.g. POApprovalExtended or leave it as proposed by SAP Web IDE ‘MM_PO_APVExtension’ Then click **Next**.

7. Click **Finish**. An extension project is created in your workspace.

8. If authentication is required, enter User Name and PW for the SAP Gateway system.
9. The application **Preview Pane** automatically opens in **Preview Mode**. Once the application is loaded and the data has been fetched from the backend, you can see that the application is fully running.

<table>
<thead>
<tr>
<th>Search</th>
<th>Purchase Orders (11)</th>
<th>Purchase Order</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Electronic Components... 1,194.00</td>
<td>Electronic Components Distributor 1,194.00</td>
</tr>
<tr>
<td></td>
<td>3 items</td>
<td>Yesterday</td>
</tr>
<tr>
<td></td>
<td>Christian Holzworth</td>
<td></td>
</tr>
</tbody>
</table>

10. Explore the features of the preview.
   a. Display the app in the different screen sizes
   b. When in Small size, change the orientation
   c. At the end, go back to Large or Medium size
11. In the application Outline, select the Show property **Extensible Elements** in the drop down list.

12. Toggle to **Extensibility Mode** by clicking **Preview Mode** down arrow. Notice that the app gets a shadow. Hover the mouse of the controls in the app. Notice how views are highlighted and the scrolling occurs automatically.
### 4.3 Extend the app by hiding an existing control

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. In the app, navigate to the view S4 by scrolling down in the detail view until you see the Items table, click on an item.</td>
<td><img src="image1.png" alt="Screenshot 1" /></td>
</tr>
<tr>
<td>Note: You may need to toggle off Extensibility Mode, because in Extensibility Mode you cannot navigate to other views.</td>
<td></td>
</tr>
<tr>
<td>2. We are going to hide the Pricing Conditions in the view S4.</td>
<td><img src="image2.png" alt="Screenshot 2" /></td>
</tr>
<tr>
<td>Make sure you are in Extensibility Mode. Select the Pricing Condition at the bottom of the view. To highlight the control you need to point the mouse pointer to the right or left side of the table. Alternatively, you may select in the Outline the PricingCondTable under S4.</td>
<td></td>
</tr>
<tr>
<td>3. Select Hide Control in the drop-down of Possible Extensions and click OK to hide it - busy indicator appears, followed by a popup message saying the extension was added successfully. Alternatively, you can right-click on PricingCondTable in the Outline and select Hide Control</td>
<td><img src="image3.png" alt="Screenshot 3" /></td>
</tr>
</tbody>
</table>
4. A pop-up window appears to indicate that the control is successfully hidden. Click **Yes** to refresh the application.

5. Navigate to **S4** view and show the control was hidden

6. In Outline section, under **Show** field, drop down to select **Extended Elements**. Now, you can see that under **S4** element **PricingCondTable** is listed as Hidden.

7. Close the Extensibility Pane by clicking on the X on the top right
### 4.4 Extend the app by adding a new UI field to an extension point

<table>
<thead>
<tr>
<th><strong>Explanation</strong></th>
<th><strong>Screenshot</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Highlight the application folder and start the Extensibility Pane (Tools &gt; Extensibility Pane or Ctrl-Shift-E) if not done yet.</td>
<td><img src="image1.png" alt="Screenshot 1" /></td>
</tr>
<tr>
<td>2. In the application Outline section, select the Show property Extensible Elements in the drop down list.</td>
<td><img src="image2.png" alt="Screenshot 2" /></td>
</tr>
</tbody>
</table>
3. Expand S2 view in the Outline and select the extension point `extListItemInfo`.

4. Check that the possible extension at the bottom of the Application Outline shows `Extend View` and click `OK`.
5. Show that the extension point was successfully extended and click Yes to refresh the application.

6. Check to see that the element extListItemInfo displays (Extended). Select it and click Go to Code, Extension Code at the bottom.
7. You are redirected back to the Code Editor and the generated extension fragment file is opened.

8. Paste the following code:

```xml
<ObjectAttribute
    text="{i18n>view.PurchaseOrder.
        purchaseOrderLabel}: {PoNumber}"/>
```

9. The filename has a prefixed * to indicate that the file has been edited and not saved. Click Save.

10. Run the app in the Extensibility Pane (Tools > Extensibility Pane)

11. You can now see the added purchase order number and the label in English. The value of the purchase order has been fetched from the backend. Check the detail view to verify the same purchase order number.
12. In the top bar, click on the down arrow next to the language English. Select another language, e.g. French.

13. Show that the language of the label has changed according to the language selected

14. Repeat for other languages. At the end, you may go back to English.
## 4.5 Extend the app by replacing a view

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select <strong>S3</strong> view (detail view) in the Outline. Notice the icon tab bar and remember the number of icons you can see. In the case of the screenshot, there are 2 icons (information and notes). Other POs may have other icons shown, depending on the PO data.</td>
<td></td>
</tr>
<tr>
<td><img src="image.png" alt="Screenshot" /></td>
<td></td>
</tr>
<tr>
<td>2. Select <strong>Replace with a copy of the parent view</strong> in the drop-down of Possible Extensions. Click <strong>OK</strong>.</td>
<td></td>
</tr>
<tr>
<td><img src="image.png" alt="Screenshot" /></td>
<td></td>
</tr>
</tbody>
</table>
3. On the success message, click **No**. There is no need to refresh the application at this moment, as we want to apply some changes to the code.

4. Check the application Outline to validate that the S3 view now has the indication of (Replaced with S3Custom)
5. Select S3 view in the Outline (if not already selected) and click Go to Code at the bottom. You are redirected back to the Code Editor and the copied view file opens.

6. You are redirected back to the Code Editor and the copied view file opens.

7. We will be adding more icon tabs to the tab bar. The file on the right contains the extension we want to apply. Open the file and copy the code.
8. Paste the code to the S3Custom.view.xml file as follows:
At line 31, you can find multiple `<IconTabFilter> items. Paste all the code before the `<IconTabFilter> items.

9. Click **Save** to save the file.

10. Select **index.html**, click **Run** to run the app.
11. Show the result.

In this particular PO, we have 2 additional icons that have been added.

At the end, you may close the Application Preview browser tab.
4.6  **Extend the existing OData service in Gateway Service Builder**
(These steps are described in the guide “Creating a ‘Fiori Like’ Application - SO Tracking”)

4.7  **Replace the OData Service by an extended OData Service**

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select your project folder and start a new wizard to replace the service (<em>File &gt; New &gt; Extension</em>)</td>
<td><img src="image1.png" alt="Screenshot 1" /></td>
</tr>
<tr>
<td>2. Make sure the right project name is displayed in the text field. Click <em>Next</em></td>
<td><img src="image2.png" alt="Screenshot 2" /></td>
</tr>
</tbody>
</table>
3. Click on **Replace Service** to select the OData Service replacing wizard, and then click **Next**.

4. At wizard step 3 (Data Connection), click on **Service Catalog** to get all available OData Services on the SAP Gateway system.

   Click the drop down list and select the Gateway system (**GM0**). The list of OData Services is being fetched.

   Scroll down the list to display all OData Services available on the Gateway system selected.
5. Search for "poapproval" to display search and filter capabilities.

Expand the OData Service and show defined data collections for the OData Service ZGBAPP_POAPPROVAL_DEMO_SRV is shown here.

Select the relevant OData Service, ZGBAPP_POAPPROVAL_DEMO_SRV.

6. Scroll down to WorkflowTaskCollection and expand it
7. Scroll down to the end of the WorkflowTaskCollection and show that the new field \textit{DemoExtension} has been added.

8. Go back up to the list, select \texttt{ZGBAPP\_POAPPROVAL\_DEMO\_SRV} and click \textit{Next}.

9. At wizard step 4, click \textit{Finish} to confirm adding the extension.
16. We will now add a text field to the UI to display the extended OData field.

Double-click the file `extListInfoCustom.fragment.xml` in the view folder.

17. Paste the following code:

```xml
<ObjectAttribute text="OData Extension: {DemoExtension}"/>
```

18. The filename has a prefixed * indicating the file has been edited and not saved. **Save** the file.

19. Select the index.html file and **Run** the application.

20. You can now see the new OData field.
### 4.8 Extend the data model using the EDMX editor, then add a reference to the new data property in the UI logic

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Navigate to the project &gt; <strong>model</strong> folder. You may want to close all other tabs.</td>
<td><img src="image1.png" alt="Screenshot 1" /></td>
</tr>
<tr>
<td>2. Double-click on the file <strong>metadata.xml</strong> to open the file in the editor</td>
<td><img src="image2.png" alt="Screenshot 2" /></td>
</tr>
<tr>
<td>3. Locate the EntityType with Name=&quot;<strong>HeaderDetail</strong>&quot; You may use the search with Ctrl-f and enter the workflowtask in the search field.</td>
<td><img src="image3.png" alt="Screenshot 3" /></td>
</tr>
</tbody>
</table>
4. Add a new property to this EntityType:

```xml
<Property Name="NewField" Type="Edm.String" Nullable="false" MaxLength="10" sap:label="New Field"/>
```

5. Save the `metadata.xml` file.

6. Now, we will be adding a UI text field in the detail view to display the added New Field.

Open the file with the extension fragment in the view folder: `S3Custom.view.xml`

Search for `NewField` by pressing Ctrl-f and enter 'newfield'
7. Uncomment the block that contains the New Field definition by selecting the commented block.

8. **Edit > Comment > Toggle Block Comment**

9. **Save** the file.
10. Now we will run the app with Mockdata to demonstrate that the new field is displayed by using the extended metadata.xml file.

Select the file `index.html` of the application. Run the app (Run > Run with Mockdata).

11. Ensure that the URI is `/sap/opu/odata/ZGBAPP_PO_APPROVAL_DEMO_SRV` and Mock Data Source is ‘Random data’:
   - Select project name, right mouse click, select Project Settings.
   - Click on Mock Data, verify Mock Data Source.

This will allow the system to generate data for the different data fields.
12. Verify that data has been generated.
### 4.9 Deploy the application to SAP Gateway
(Note: It could also be deployed to HANA Cloud Platform)

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Go back to SAP Web IDE. You may want to close other open tabs and windows.</td>
<td><img src="image1" alt="Screenshot" /></td>
</tr>
<tr>
<td>2. Select the project, right-click and select <strong>Deploy &gt; Deploy to SAPUI5 ABAP Repository</strong>.</td>
<td><img src="image2" alt="Screenshot" /></td>
</tr>
<tr>
<td>3. For the Deployment Options, select the system GM0</td>
<td><img src="image3" alt="Screenshot" /></td>
</tr>
</tbody>
</table>

**Deploy to SAPUI5 ABAP Repository**

1 Deployment Options
4. If authentication is required, enter User Name and PW for the SAP Gateway system.

5. Keep **Deploy a new application** selected and click **Next**.

6. Enter a name, e.g. Z_POAPP00 (the name needs to start with Z), enter any description and click **Browse**.

   Note: In Package field, $TMP is defaulted by the system if your system only support local object creation.

   Click **Next**.

7. Click **Finish** to deploy your application to the SAPUI5 ABAP Repository.
8. Wait for the successful feedback. Click on **OK**
### 4.10 Verify successful deployment of the app
(by importing the app from SAP Gateway to SAP Web IDE)

<table>
<thead>
<tr>
<th>Explanation</th>
<th>Screenshot</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Select the <strong>Local</strong> Folder at the very top of the project list and create a new folder (right-click <strong>Local New &gt; Folder</strong>)</td>
<td><img src="image1" alt="Create Folder" /></td>
</tr>
<tr>
<td>2. Provide a name, e.g. <strong>POApprovalExtendedImported</strong>, and click <strong>Create</strong>.</td>
<td><img src="image2" alt="Create Folder" /></td>
</tr>
<tr>
<td>3. Right-click the new folder and select <strong>Import &gt; Application from SAPUI5 ABAP Repository</strong></td>
<td><img src="image3" alt="Application from SAPUI5 ABAP Repository" /></td>
</tr>
</tbody>
</table>
4. Select the system where the app had been deployed to (GM0)

5. Search for the app, e.g. Z_POAPP00
6. Select the app and click **OK**.

7. A pop-up window message indicates the import has finished. Click **OK**.

8. Run the application and verify the application is running
   1. Select the file **index.html** of the application
   2. Click on the **Run**

9. If authentication is required, enter User Name and PW for the SAP Gateway system.
10. The Application Preview is started, the application is loading and data is fetched from the OData Service.