

SAP Mobility

SAP Project Fiori Program

# Extending SAP Fiori Apps

*A Branded Service provided by SAP Rapid Innovation Group*

**Applicable Releases:**

**SAP Project Fiori Program**

**Version 1.3**

**July 2013**



The Best-Run Businesses Run SAP™

© Copyright 2013 SAP AG. All rights reserved.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects Software Ltd. Business Objects is an SAP company.

SAP "How-to" Guides are intended to simplify the product implementation. While specific product features and procedures typically are explained in a practical business context, it is not implied that those features and procedures are the only approach in solving a specific business problem using SAP NetWeaver. Should you wish to receive additional information, clarification or support, please refer to SAP Consulting.

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.

#### Disclaimer

Some components of this product are based on Java™. Any code change in these components may cause unpredictable and severe malfunctions and is therefore expressly prohibited, as is any decompilation of these components.

Any Java™ Source Code delivered with this product is only to be used by SAP's Support Services and may not be modified or altered in any way.



The Best-Run Businesses Run SAP™

## Document History





Document Version	Description
1.0	First release version
1.1	Added Appendix A: Testing an SAP Fiori App locally
1.2	Various Improvements
1.3	Added Appendix B: Internal Structure of SAP Fiori Apps



## Typographic Conventions

Type Style	Description
<i>Example Text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbutton labels, menu names, menu paths, and menu options. Cross-references to other documentation
Example text	Emphasized words or phrases in body text, graphic titles, and table titles
Example text	File and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
<b>Example text</b>	User entry texts. These are words or characters that you enter in the system exactly as they appear in the documentation.
<b>&lt;Example text&gt;</b>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

## Icons

Icon	Description
	Caution
	Note or Important
	Example
	Recommendation or Tip

## Table of Contents

1.	Getting started.....	1
2.	Downloading Fiori App Source Code .....	2
3.	Importing Fiori App Source Code into Eclipse.....	5
4.	Modifying a SAP Fiori App's Source Code .....	9
5.	Uploading customized SAP Fiori App Code.....	13
6.	Testing your Application.....	17
	Appendix A: Testing an SAP Fiori App locally .....	19
	Appendix B: Internal Structure of SAP Fiori Apps.....	22

# 1. Getting started

This how-to guide will introduce you to modifying SAP Fiori Apps. Throughout the guide, you will learn how to download, change, re-upload, and test an SAP Fiori App.

The example used for this guide is the Purchase Approval scenario. Throughout this guide, said scenario will be extended so the end user can also see the *“Purchasing Group”* when viewing the details of a Purchase Order.

## 1.1 Prerequisites

- You should have an Eclipse installation which has the latest version of the *SAPUI5 Application Development* feature installed. This could be your own (custom) Eclipse installation or the readily available SAPUI5 Eclipse.
- You should be aware of your present SAP Fiori system architecture. This includes knowing which systems are involved, i.e. the target SAP ERP and NetWeaver Gateway component.
- The how-to guide *Adding custom fields to SAP Fiori apps in 3 steps / oData fields extensibility* has been completed for your SAP Fiori system.

## 2. Downloading Fiori App Source Code

The first step you need to take for modifying a SAP Fiori App is to download the original Web App. For this step, you will need to know which system the SAP Fiori Web Apps are hosted on. In this present guide, we will assume that the Web Apps in question are hosted on the SAP system which also hosts SAP NetWeaver Gateway.

For convenience, the *NetWeaver UI Add-On (UI\_INFRA)* includes the executable program **UI5/UI5\_REPOSITORY\_LOAD** which can be used to download, upload, and delete SAPUI5 applications.




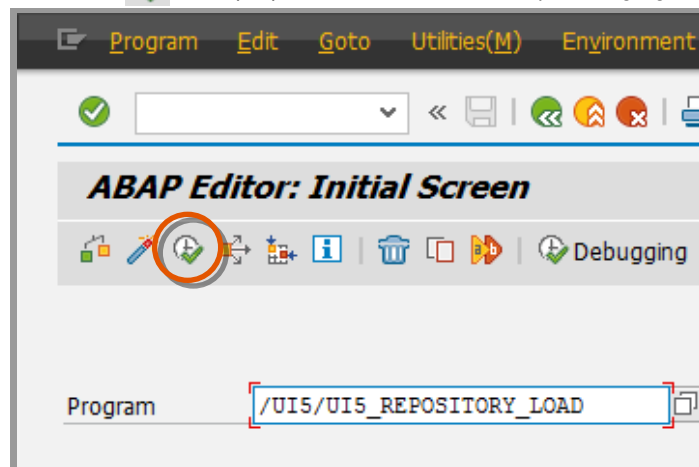
### Tip:


When a SAPUI5 application is hosted on an SAP system, it is typically uploaded as a Business Server Page. As a consequence, you can also access the SAPUI5 application files using WebDav. In addition, you can edit files directly in the *ABAP Workbench* (transaction **SE80**) or any ABAP-enabled Eclipse installation. However, none of these options will be covered any further in this How-To Guide.

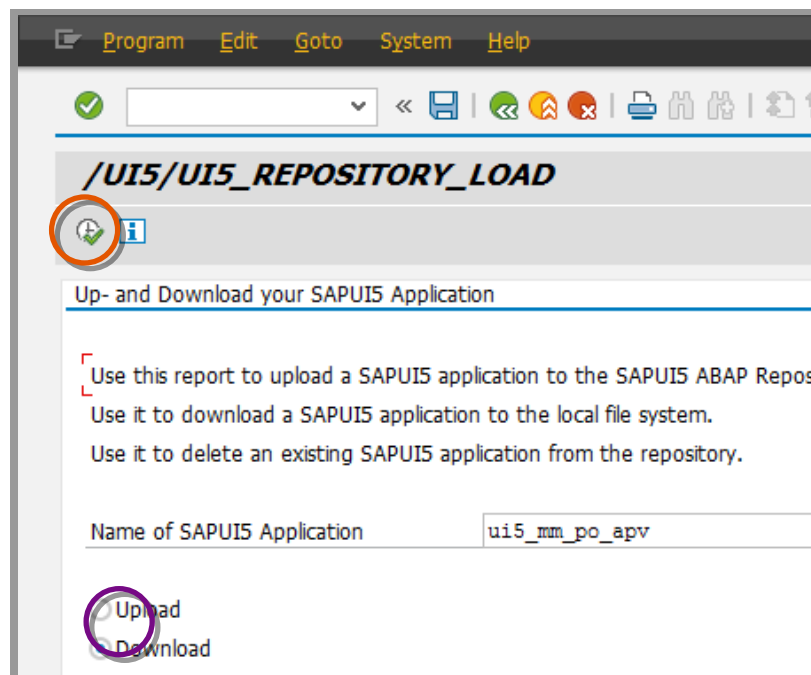


### Task:

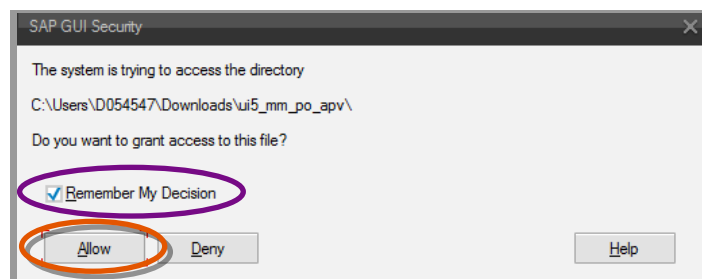
1. Logon to the appropriate SAP system and go to transaction **SE38**.
2. Enter **/UI5/UI5\_REPOSITORY\_LOAD** in the *Program* field and click  **Run (F8)** to run the SAPUI5 repository synchronization program.



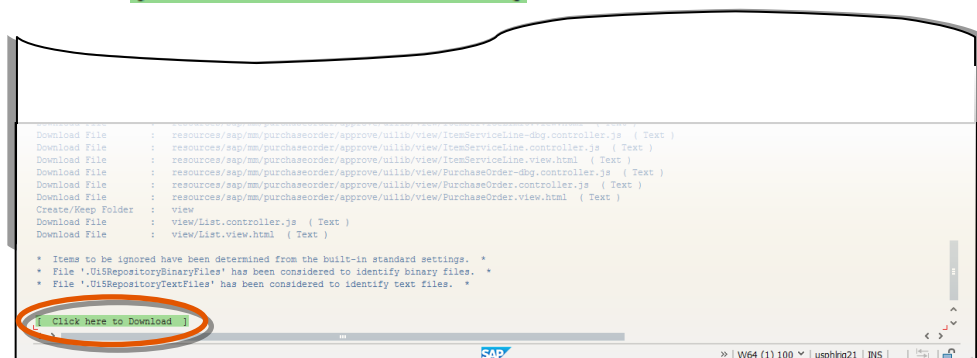
3. Next, specify the SAPUI5 application you wish to download. In order to download the SAP Fiori Web App for purchase order approval, use **ui5\_mm\_po\_apv** as *Name of SAPUI5 Application*. Ensure that the *Download* option is selected. Click  **Run (F8)**.




4. You will be prompted to select the local directory into which the Web App source files should be saved. Please ensure that the directory you chose is empty. Any files contained in the folder may be deleted before the download process starts. Choose and confirm a target directory.
5. Depending on your system configuration, you may see a security prompt. Grant permanent file system access by checking *Remember My Decision* and clicking *Allow*.

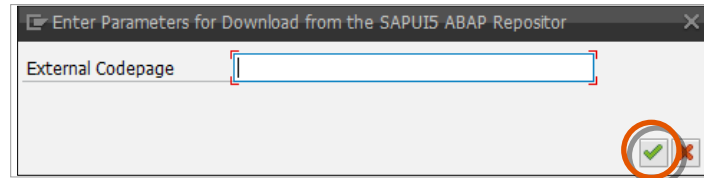


6. Next, you will see a summary of actions that are about to be taken. Scroll down and click **Click here to Download**.

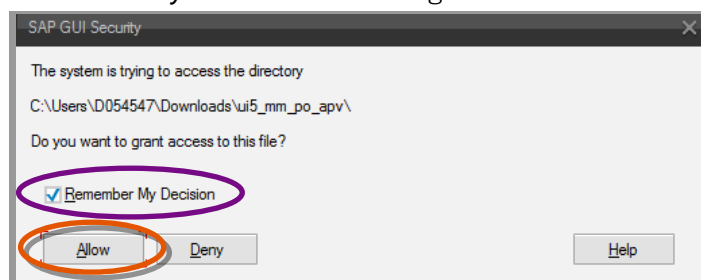




7. You will be prompted to specify an *External Codepage*. Leave the field empty and confirm by clicking on the checkmark button . The download process will be initiated.



8. Depending on your system configuration, you may see another security prompt. Grant permanent file system access to the downloader by checking *Remember My Decision* and clicking *Allow*.




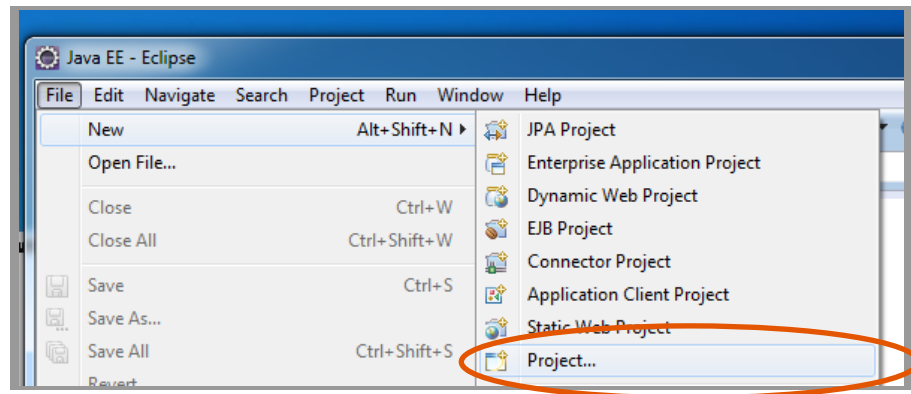
9. The download process may take a couple of minutes. When the process has finished, the SAP Frontend status bar will display a success message.



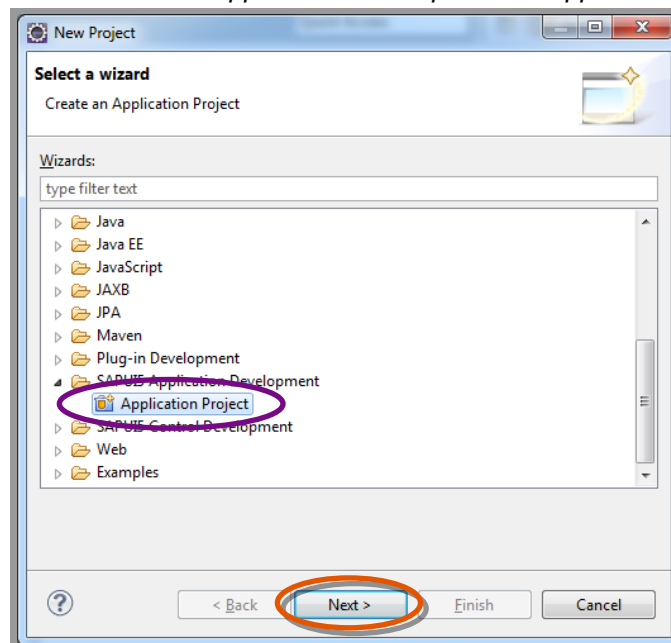
### 3. Importing Fiori App Source Code into Eclipse

In order to modify a SAP Fiori scenario, it is recommended to do so with an Eclipse installation which has the *SAPUI5 Application Development* feature installed. This simplifies the development process because you can rely on useful tools such as code completion and app preview. We will now create a new Eclipse project containing the SAP Fiori scenario source code.

-  Task: 10. Start Eclipse and select *File* → *New* → *Project...*



11. Choose *SAPUI5 Application Development* → *Application Project* and click *Next*.



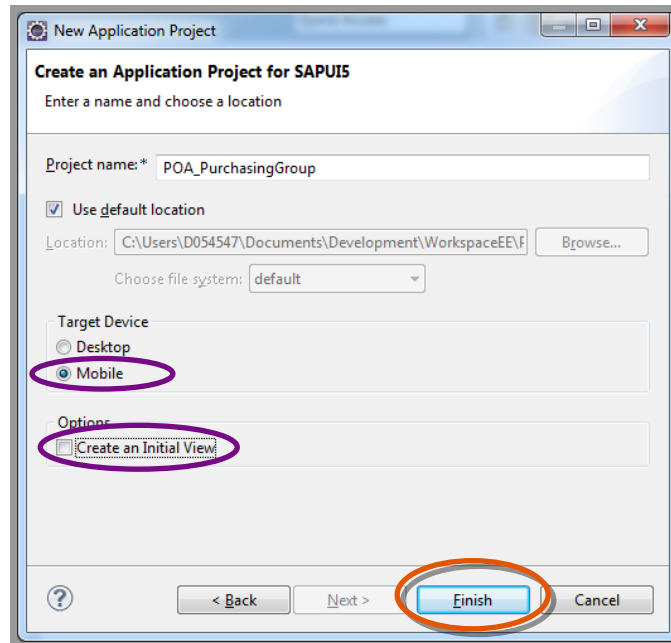
12. Next, configure the new project as follows:

*Project Name:* POA\_PurchaseGroup

*Target Device:* Mobile

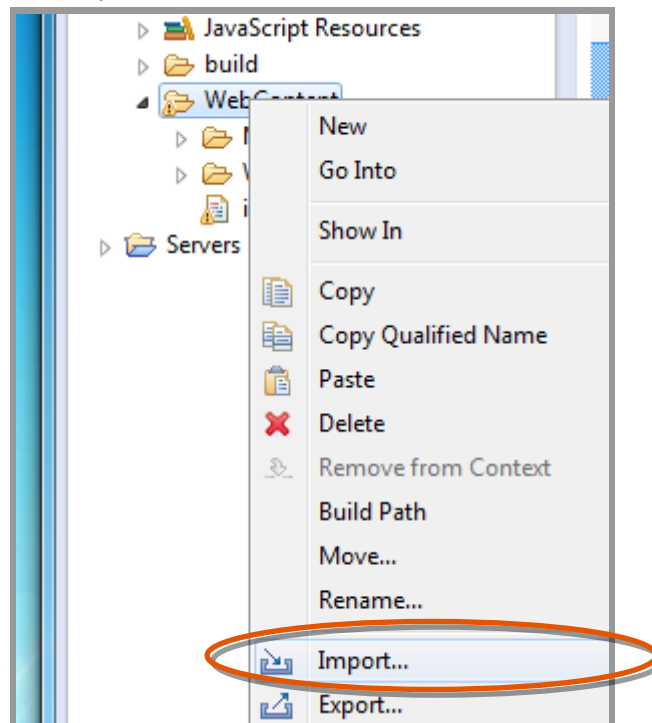
*Deselect **Create an Initial View**.*

*Click **Finish**.*

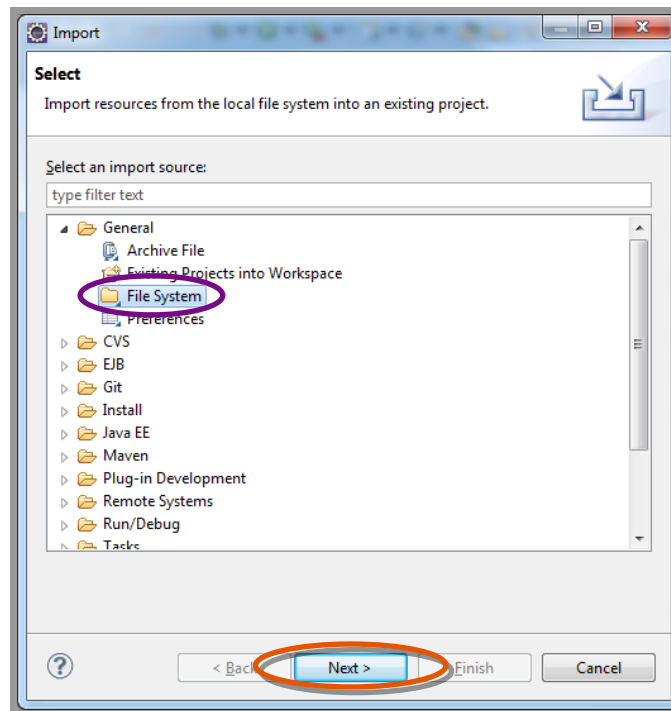


13. You should see the newly created project in the *Project Explorer*. Eclipse created the basic boilerplate SAPUI5 folder structure. We will now import the SAP Fiori source code.

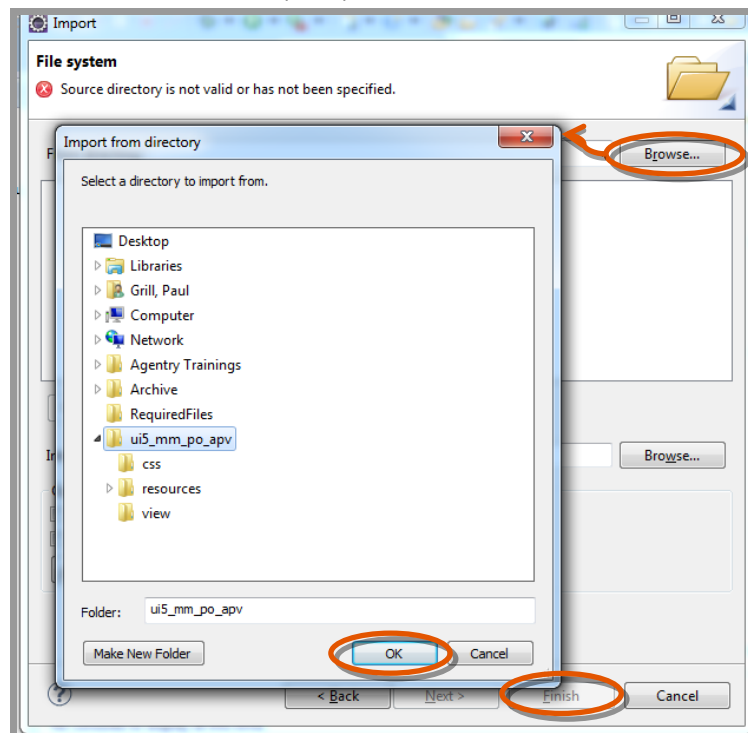
Expand the newly created project, right-click on the **WebContent** folder and click *Import...*



14. Choose *General* → *File System* and click *OK*.

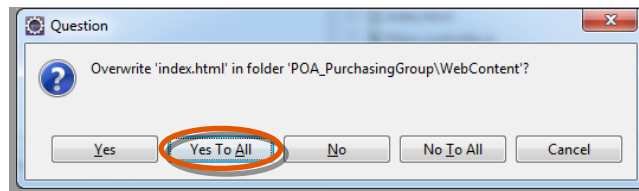


15. Select the import directory into which you downloaded the SAP Fiori App source code (refer to Task 4). Check the checkbox next to the top folder in the selection screen to ensure that all files and sub-folders are imported. Click *Finish* to initiate the import process.



16. The boilerplate SAPUI5 folder structure may contain a few files that might also exist in the SAP Fiori App source directory, e.g. `index.html`. Please click

**Yes to All** when asked whether you would like to overwrite these files.



## 4. Modifying a SAP Fiori App's Source Code

Having completed the previous steps, you now can browse and edit the source code of the SAP Fiori App you downloaded. At this point, you need to know which files to edit in order to achieve the anticipated change.



Tip:

It may take some effort to find the appropriate files to edit. Consider the following:

- Refer to Appendix B as it illustrates the internal structuring of SAP Fiori Apps further.
- Do you need to change the behavior of the app? If so, you will need to make changes to a controller.
- Do you need to change the appearance of the app? If so, you will need to make changes to a view.
- What screen segment are you making changes to? This can help you figure out the correct view/controller.
- You may also use the code search feature of Eclipse in order to find the appropriate files to edit. First, considering the area of the App's User Interface to which you would like to make changes, find a text snippet that fulfils the following requirements:
  - The text snippet appears to be static, i.e. does not come from a data source. Examples are button texts, explanation texts, field labels.
  - The text snippet is as unique as possible
  - The text snippet is close to the area you would like to make changes to as possible.

Then, ensure your project is selected in Eclipse's *Project Explorer* and press **Ctrl-H** to open *Code Search*. Switch to the File Search tab, enter your search string and start the search.

In case the text snippet you chose for searching is part of a translatable string, you will find the technical name of the translatable string. Repeat the search; this time using the technical name of the translatable string to see where it is used.

In this how-to guide, we want to add another field to the details view of a purchase order. Adding a field only requires making changes to a view, not a controller. The appropriate view is called

`sap.mm.purchaseorder.approve.uilib.component.poa.toggleArea.view.PurchaseOrderInformation`

Hence, the view in question can be found in the following location, relative to the **WebContent** folder of your Eclipse project:

`./resources/sap/mm/purchaseorder/approve/uilib/component/poa/toggleArea/view`

To add a field to the UI you have to make changes to `PurchaseOrderInformation.view.html`.

- Task: 17. Open PurchaseOrderInformation.view.html in Eclipse. Marked in Figure 1 is the depth of the **PurchaseOrderInformation** view, i.e. all fields said view covers. The **“Purchasing Group”** field that we would like to add will be placed above the **“Purchase Order ID”** field and look similar to the latter.
- Within the **PurchaseOrderInformation** view, copy the source code that yields the **“Purchase Order ID”** field and insert said code so it produces another field. As of now, you have the **“Purchase Order ID”** field twice.

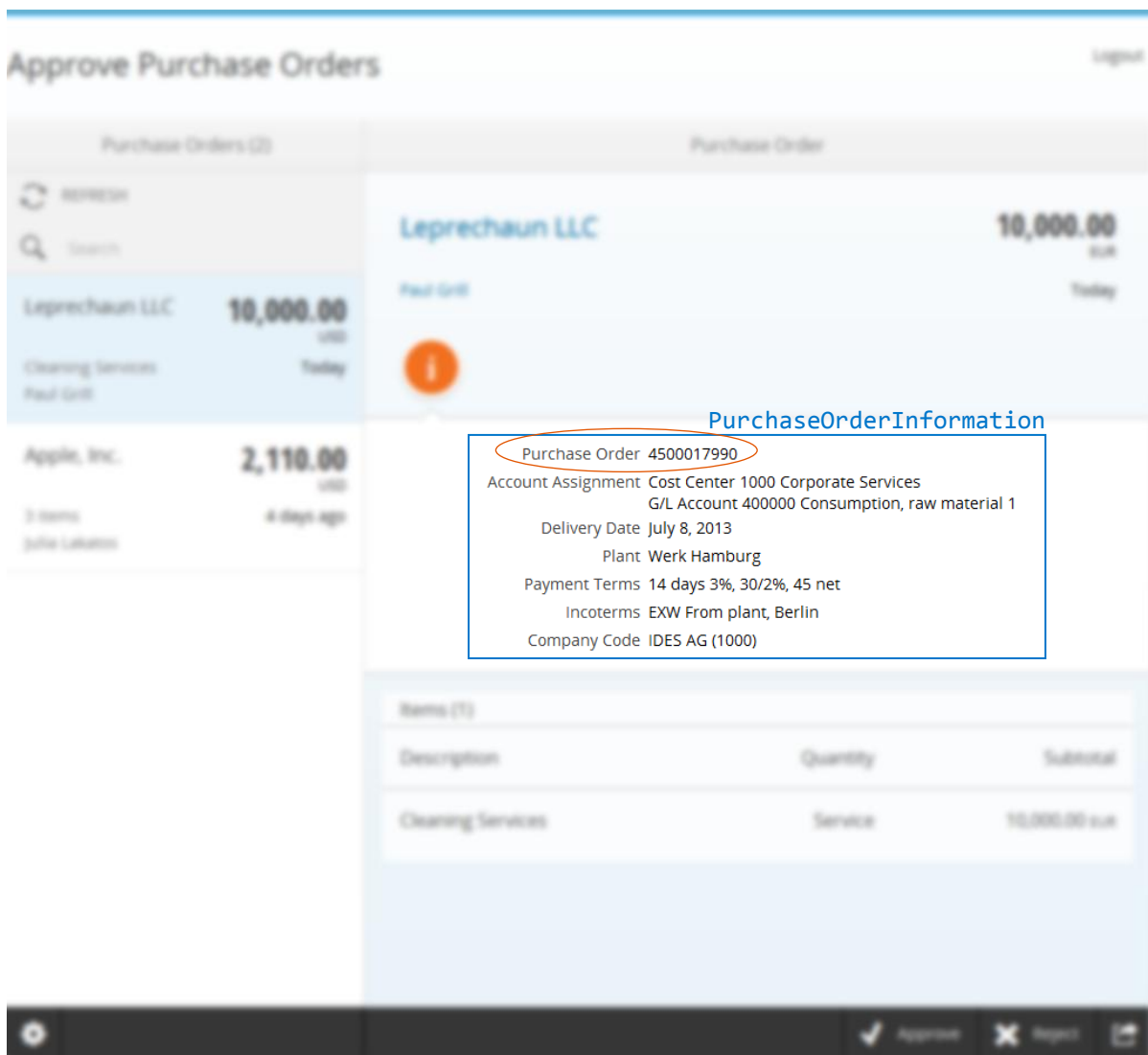


Figure 1: PurchaseOrderInformation view on Approve Purchase Order App

18. Change the newly inserted code so it will represent the Purchasing Group instead of the Purchase Order ID. Carry out the following modifications:
- Change the label text so it will read **“Purchasing Group”** in the appropriate logon language. As this is a translation-relevant task, you will have to create a new translation entry in the appropriate resource files later.

For now, change the `data-text` attribute of `sap.m.Label` to `{i18n>view.PurchaseOrder.zPurchasingGroup}`.

- Change the text view so its contents will be bound to the field in your data model which contains the Purchase Group. To do so, change the `data-text` attribute of `sap.m.Text` to `{ZPurchasingGroup}`

19. Since the term "Purchasing Group" should be translatable, it needs to be added to the internationalization (i18n) resource files instead of being written out directly in the view's source code.

Add the term `view.PurchaseOrder.zPurchasingGroup` with the appropriate translation to all i18n resource files. You may find the i18n resource files in the following location, relative to the `WebContent` folder of your Eclipse project:

```
./resources/sap/mm/purchaseorder/approve/uilib/i18n/
```

20. You can see the implementation of the new field for the purchasing group in Snippet 1. The highlighted sections indicate what has to be changed. Snippet 2 and Snippet 3 show the changes in the I18N files.
21. Local testing might not work out-of-the-box. Refer to *Appendix A* if you are interested in testing your Web App locally.

```
<div data-sap-ui-type="sap.ui.commons.form.FormElement">

  <div data-sap-ui-aggregation="layoutData"
    data-sap-ui-type="sap.ui.commons.layout.ResponsiveFlowLayoutData"
    data-linebreak="true" data-margin="false"></div>

  <div data-sap-ui-aggregation="label" data-sap-ui-type="sap.m.Label"
    data-text="{i18n>view.PurchaseOrder.zPurchasingGroup}"></div>

  <div data-sap-ui-aggregation="fields" data-sap-ui-type="sap.m.Text"
    data-text="{ZPurchasingGroup}">

    <div data-sap-ui-aggregation="layoutData"
      data-sap-ui-type="sap.ui.commons.layout.ResponsiveFlowLayoutData"
      data-weight="2" data-align-items="End"></div>

  </div>
</div>
```

Snippet 1: Additional Form Element for Purchasing Group



```
#Extra: Purchasing Group  
view.PurchaseOrder.zPurchasingGroup=Purchasing Group
```


Snippet 2: English l18N file

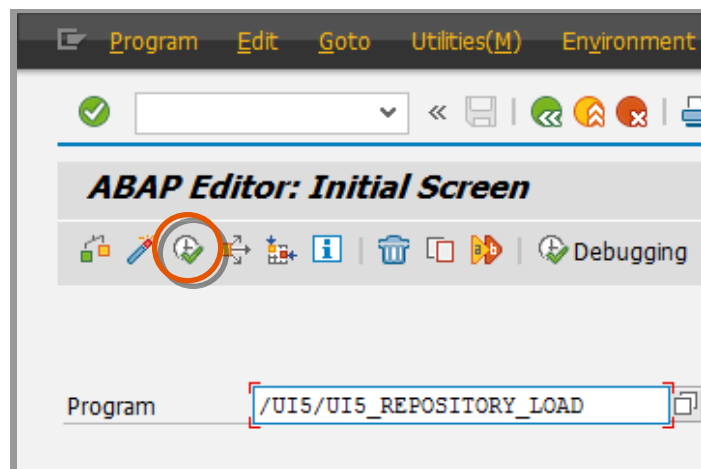
```
#Extra: Purchasing Group  
view.PurchaseOrder.zPurchasingGroup=Einkaufsgruppe
```


Snippet 3: German l18N file

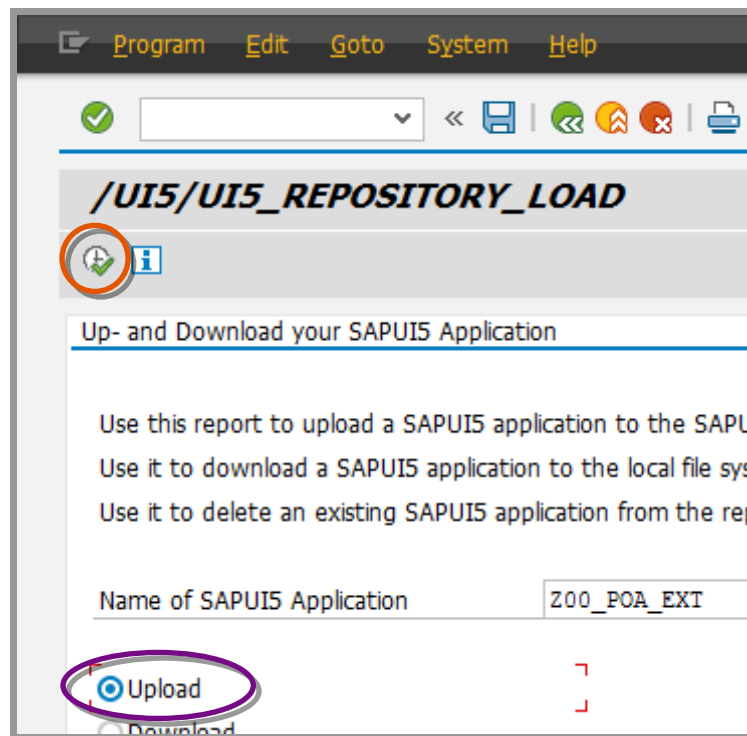
## 5. Uploading customized SAP Fiori App Code

After locally customizing a SAP Fiori App to your requirements, the next step is to upload the modified SAP Fiori App so it becomes available on the server. You can either overwrite the old application or you can specify a new location at which the end users will be able to access the new application. We will once again use the executable program `/UI5/UI5_REPOSITORY_LOAD` for the upload process.

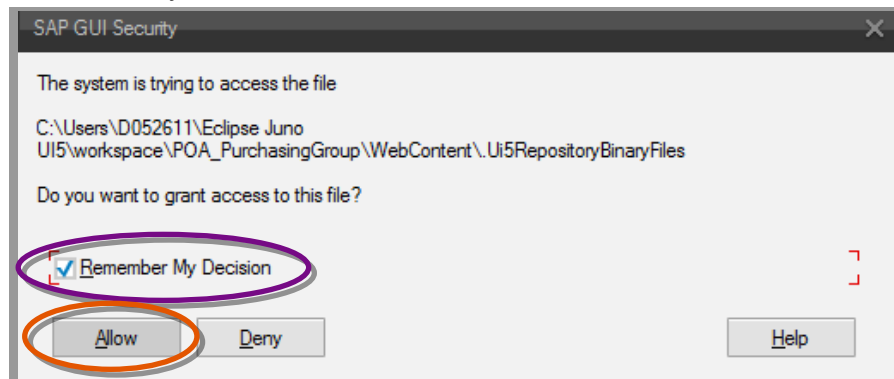
-  Task: 22. Logon to the appropriate SAP system and go to transaction **SE38**. The appropriate system is the same system from which you downloaded the source code earlier.



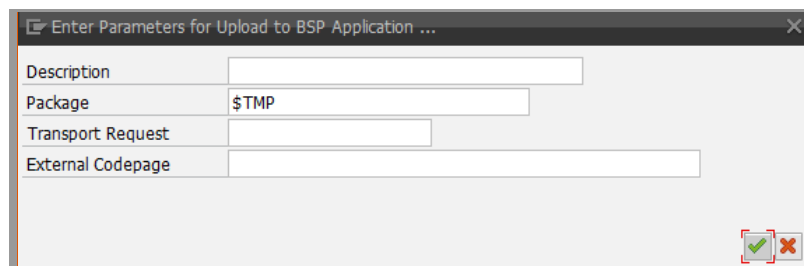
23. Next, enter a name for the application you are uploading into the field **Name of SAPUI5 Application**. Please note that there is a 15-character limit. Also, for this example, please ensure the application name begins with a **Z**. The application name will later be the name of your Business Server Application. As a consequence, the application name will also show up in the final URL. Ensure that the **Upload** option is selected. Click  **Run (F8)**.



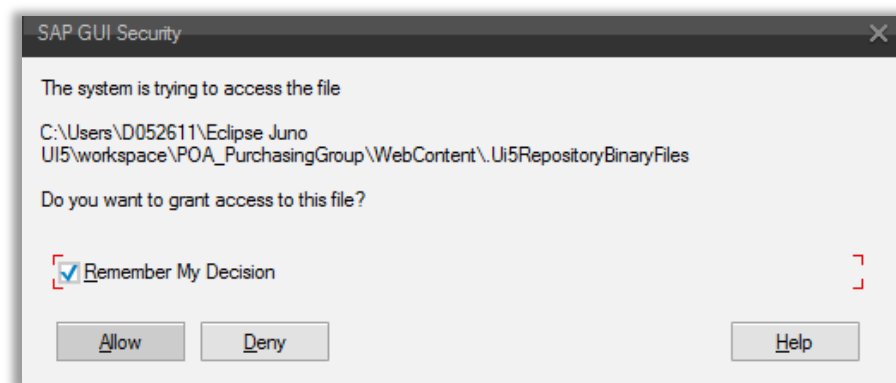
24. You will be prompted to choose the source directory for upload. Find and select the location of the **WebContent** folder of your Eclipse project.
25. You might be asked whether you want to grant access to your files. Choose *Remember My Decision* and click on *Allow*.



26. Next, you will see a summary of actions that are about to be taken. Scroll down and click [ **Click here to Upload** ].
27. Before the upload process begins, you will be asked a few additional parameters. For this example, use **\$TMP** as package name.



28. Depending on your system configuration, you may see security prompts for every file that is uploaded to the SAP ERP system. You can change this behaviour in the *SAP Frontend Security Settings*. It is highly recommended to do so.



To change the security settings, open the *SAP GUI Configuration* application. You may find this application in the “*SAP Front End*” folder in your operating system’s start menu. Open the *Security*-Tab and choose *Security Settings*. Click *Open Security Configuration*. You will see a list of files and folders SAP GUI has previously accessed on your computer. Select the entry which represents the **WebContent** folder in your Eclipse workspace with the action “Read”. Click on *Edit*. Change *Action* to *Allow* and save your changes. Also refer to Figure 2 for more information.

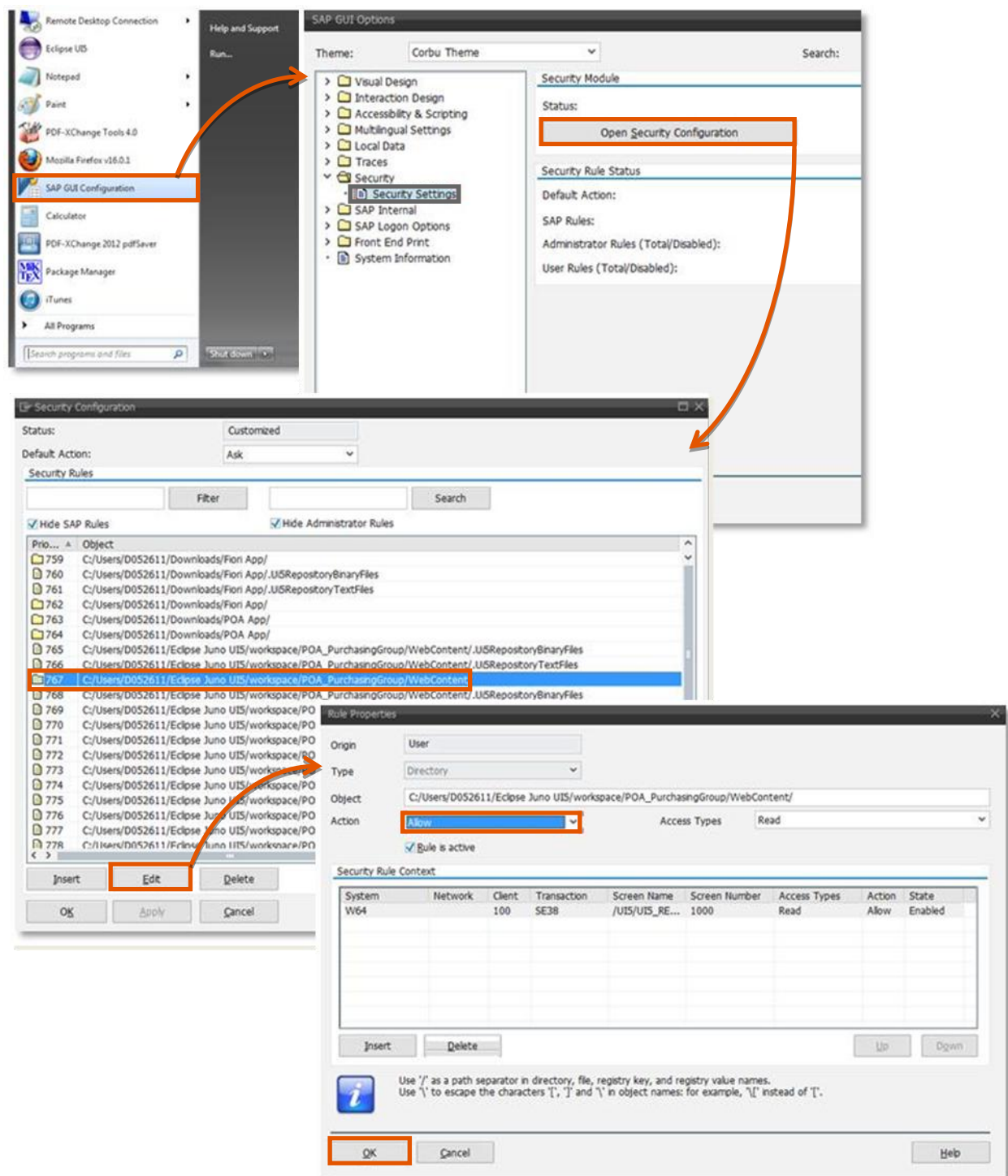

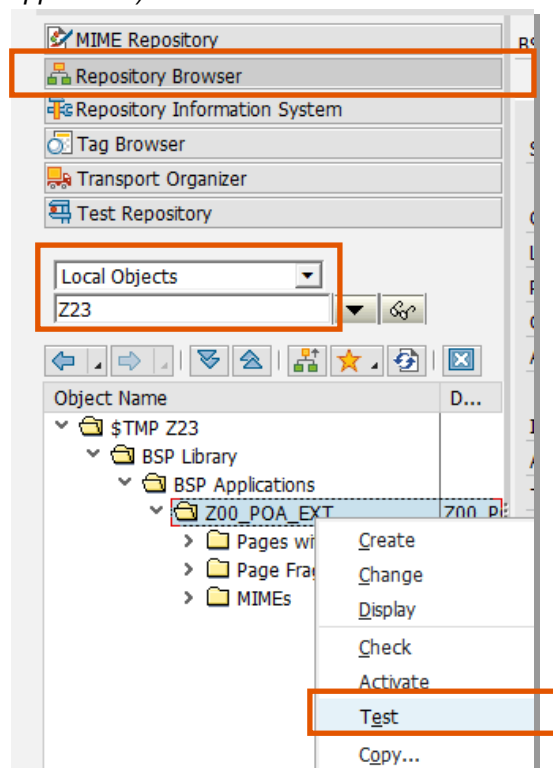


Figure 2: Changing Security Settings

## 6. Testing your Application

Now you can test your modified application.

-  **Task:** 29. On your SAP ERP system, use transaction **SE80** to navigate to the package which you specified during the upload process (refer to Task 27). In the present example, we used the local package **\$TMP**.
30. Find the application you just uploaded in the repository browser. In the object tree, you will find it under *BSP Library* → *BSP Application* → (*Name of your application*).



31. Right-click the application name and choose **Test**.
32. SAP GUI will attempt to open the application in your web browser. This may lead to another security prompt which you can safely confirm.

Your application should now feature the field “Purchasing Group” in the details view for a purchase order, just like in Figure 3.

# Approve Purchase Orders

Logout

Purchase Orders (2)	Purchase Order
<div> <div>REFRESH</div> <div>Search</div> </div> <div> <div>Leprechaun LLC</div> <div>10,000.00 USD</div> <div>Cleaning Services</div> <div>Paul Grill</div> <div>Today</div> </div>	<div> <div>Leprechaun LLC</div> <div>10,000.00 EUR</div> <div>Paul Grill</div> <div>Today</div> </div> <div> <div> <div>i</div> <div> <div>Purchasing Group Z00</div> <div>Purchase Order 4500017990</div> <div>Account Assignment Cost Center 1000 Corporate Services</div> <div>G/L Account 400000 Consumption, raw material 1</div> <div>Delivery Date July 8, 2013</div> <div>Plant Werk Hamburg</div> <div>Payment Terms 14 days 3%, 30/2%, 45 net</div> <div>Incoterms EXW From plant, Berlin</div> <div>Company Code IDES AG (1000)</div> </div> </div> </div>
<div> <div>Apple, Inc.</div> <div>2,110.00 USD</div> <div>3 items</div> <div>Julia Lakatos</div> <div>4 days ago</div> </div>	

Items (1)

Description	Quantity	Subtotal
Cleaning Services	Service	10,000.00 EUR


⚙️

✓ Approve

✗ Reject

🔗

Figure 3: Customized version of Purchase Order Approval featuring "Purchasing Group" field

 **Note:** For a better debugging experience, append the URL parameter `sap-ui-debug=true` to your SAP Fiori Web App's URL. SAP UI5 will then load the uncompressed libraries; these are bigger in size but are easier to debug.

## Appendix A: Testing an SAP Fiori App locally

While implementing changes in the source code of a SAP Fiori Web App, you might want to test your newest code locally without re-uploading it to the SAP NetWeaver Gateway server. This chapter will instruct you on how to set up your development environment for local testing.

In general, testing a SAPUI5 Web App locally is not a difficult task. However, for SAP Fiori Web Apps there is an obstacle to keep in mind: By default, every application will assume that it is currently hosted on the Gateway system from which it should also pull data. As a consequence, Fiori Apps will attempt to pull data from the domain they are hosted on. This will not work when testing locally because the domain name resolves to your development workstation, not the Gateway server.

SAP UI5's Eclipse tooling incorporates a proxy module which allows to proxy between a remote server and your PC's local domain. You can configure the proxy module to redirect any requests to SAP NetWeaver Gateway. During the following tasks, you will configure your Eclipse-based development environment so you can test SAP Fiori Web Apps locally with data from a remote SAP NetWeaver Gateway server.



### Note:

While browsing the source code of a SAP Fiori Web App, you might discover that some Fiori apps offer local testing capabilities through mock data or by redirecting OData calls to another server. We will not make use of these features as they are not consistently available across all Fiori apps. The present how-to guide will show you a method for local testing that works with most Fiori Apps.

We will also not make use of SAPUI5's Web App Preview feature.



### Task:

1. You will need to set up an Apache Tomcat web server on which your SAP Fiori App will be executed locally. In Eclipse, create a new server by navigating to *File* → *New* → *Other...*, then choosing *Server* → *Server*, and clicking *Next*.
2. Select *Apache* → *Apache Tomcat v7.0 Server* as server type, and apply the following configuration:  
  
*Server's host name:* 127.0.0.1  
*Server name:* Testing Environment for (Project Name)  
*Click **Next** to continue.*
3. If the next step of the project wizard prompts you for a Tomcat installation directory, this means that you currently do not have a local installation of Apache Tomcat v7.0. Select a directory into which you would like to install Apache Tomcat, for example *C:\Tomcat70\*. Then click on *Download and Install*.
4. After you have accepted the licensing conditions of Apache Tomcat, Eclipse will attempt to automatically download and unpack Apache Tomcat in the installation directory you specified. This process will occur in the background.



If the download process fails, download Apache Tomcat manually and unpack Apache Tomcat into the installation directory. You may download Apache Tomcat 7.0 from the following web site:

<http://tomcat.apache.org/download-70.cgi>

(Click on “**zip**” under Binary Distributions → Core)

Eclipse will let you continue once the installation directory contains an Apache Tomcat installation. Ensure that the installation directory that you specified contains the Apache Tomcat installation in such way so said directory contains the sub-folders **bin**, **conf**, **lib**, **logs**, **temp**, **webapps**, and **work**. Click on *Next* to continue.

5. In the step “Add and Remove” of the server creation wizard, select your current SAP Fiori UI5 project and click *Add* so it is configured to be run on your newly created Tomcat server.
6. Click *Finish*.
7. In Eclipse, open the *Servers* view. If you cannot find said view, navigate to *Window* → *Show View* → *Other...*, choose *Servers* → *Server*, and click *OK*.
8. Double-click on your newly created Tomcat server to open its configuration view.
9. In the configuration view, switch to the *Modules* tab.
10. In the *Web Modules* sections of the *Modules* tab, select your project and click on *Edit...*
11. Clear the field *Path* so it is empty and click *OK*.
12. Save and close the Tomcat configuration.
13. In *Project Explorer*, expand your current SAP Fiori UI5 project and double-click on *Deployment Descriptor: (SAP Fiori UI5 project name)*.
14. Switch to the *Source* tab and look for the UI5 proxy servlet section.
15. Look for the line that reads as follows:

`<url-pattern>/proxy/*</url-pattern>`

Change said line to match the following snippet:

`<url-pattern>/sap/*</url-pattern>`

16. After `</servlet-mapping>`, add the following snippet, and fill in the correct hostname and port for your SAP NetWeaver gateway system:

```
<context-param>
  <param-name>com.sap.ui5.proxy.REMOTE_LOCATION</param-name>
  <param-value>http://[hostname]:[port]/sap/</param-value>
</context-param>
```

17. In *Project Explorer*, right-click on your current SAP Fiori UI5 project, and choose *Run As* → *Run On Server*.
18. Expand the node named *127.0.0.1* and find your new Tomcat installation. Be sure to check *Always use this server when running this project*. Click *Finish*.
19. Eclipse will now attempt to start Apache Tomcat. If you encounter a “*Port already in use*” error message, go back to the *Servers* view and stop any Tomcat server instances that are already running.

As long as Apache Tomcat is running, any changes made to your project files will automatically be pushed to your local server. To restart the testing environment for your project, right-click on your project in *Project Explorer*, and choose *Run As* → *Run On Server*.

It is recommendable to repeat all steps for every SAP Fiori UI5 Eclipse project, i.e. you create a separate Apache Tomcat server for every project. When doing so, you will only have to download and install Apache Tomcat once.

## Appendix B: Internal Structure of SAP Fiori Apps

This chapter will explain the basic internal structuring of SAP Fiori Apps, i.e. the hierarchy of folders and files in the SAPUI5 application project. The objective of this section is to help you find the correct files for adjusting a SAP Fiori App to your needs.



### Note:

You will experience that every SAP Fiori App looks different internally. In general, it is important to differentiate between the following two groups:

- Apps that feature an approval process, e.g. Travel Request Approval.
- Apps that do not feature an approval process, e.g. Travel Request Creation.

The first group of Apps has the approval process in common, i.e. Apps in this group share some functionality and thus follow an additional set of conventions. These are discussed in a later section of this chapter.

## The Application

The Application component refers to the source code that is specific to a certain SAP Fiori App. This encompasses all files in the App's root folder and all its sub-folders, excluding **/resources**.

If the SAP Fiori App you are modifying does not feature an approval process, the files you need to edit will most likely belong to this component.

In general, it is important to remember that the basis of every SAP Fiori App is always its **index.html** file. When the URL of a SAP Fiori App is opened in a Web Browser, **index.html** is the file that is ultimately requested from the Gateway server. It contains the SAPUI5 "bootstrap" which references all theming and scripting files that should be loaded at startup. Additionally, it includes the JavaScript code to initialize the SAP Fiori Application. This is usually a simple call to a JavaScript object residing in another file.



### Tip:

Your first activity could be to establish the execution graph for application start-up. You will get familiar with the interaction between the different files contained in the SAP Fiori App. While doing so, make note of the namespaces that are defined with calls looking similar to the following snippet:

```
jQuery.sap.registerModulePath('namespace', 'path');
```

SAP Fiori Apps use namespacing heavily. As a consequence, it is a good idea to be aware of the path a specific namespace resolves to.



### Tip:

Another activity is to get familiar with the view files contained in the Application component. First, find the outermost view, and then analyse which other views it references. A view reference in HTML-based SAPUI5 views will always look similar to the following snippet:

```
<div
  data-sap-ui-type="sap.ui.core.mvc.HTMLView"
  data-view-name="Name of view"
  ...>
</div>
```

## The Cross-Application (CA) Library

The Cross-Application component contains coding intended for re-use. This includes UI controls such as Business Card, and dialogs such as Account Assignment and Forwarding. The CA library files are located at `/resources/sap/ca/common/uilib/`; hence, their namespace is `sap.ca.common.uilib.*`.

Although the Cross-Application library is intended for re-use, every SAP Fiori App comes with its own copy of the CA library files. As a consequence, you can safely make changes to the CA library files of one App without affecting other Apps.



### Caution:

The CA library is a SAPUI5 Library. At deployment time, SAPUI5 libraries are usually packaged and optimized so they load faster when being used in a productive environment.

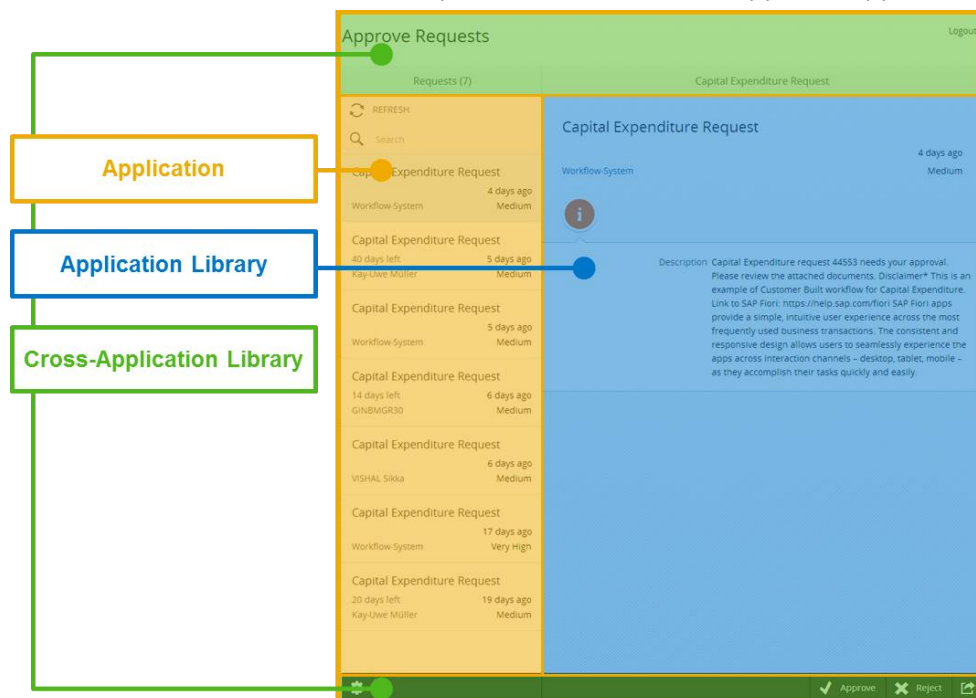
During packaging, the original JavaScript files are renamed so that they have the file ending `-dbg.js`. At runtime, SAPUI5 will only use the `-dbg.js` files if the Web App is running in debug mode, i.e. when the URL parameter `sap-ui-debug` has been set to `true`.

To conclude: Making changes to JavaScript files contained in a SAPUI5 library may not show any effect in the production environment. However, you can still make changes to all other files contained in the library, e.g. HTML files.

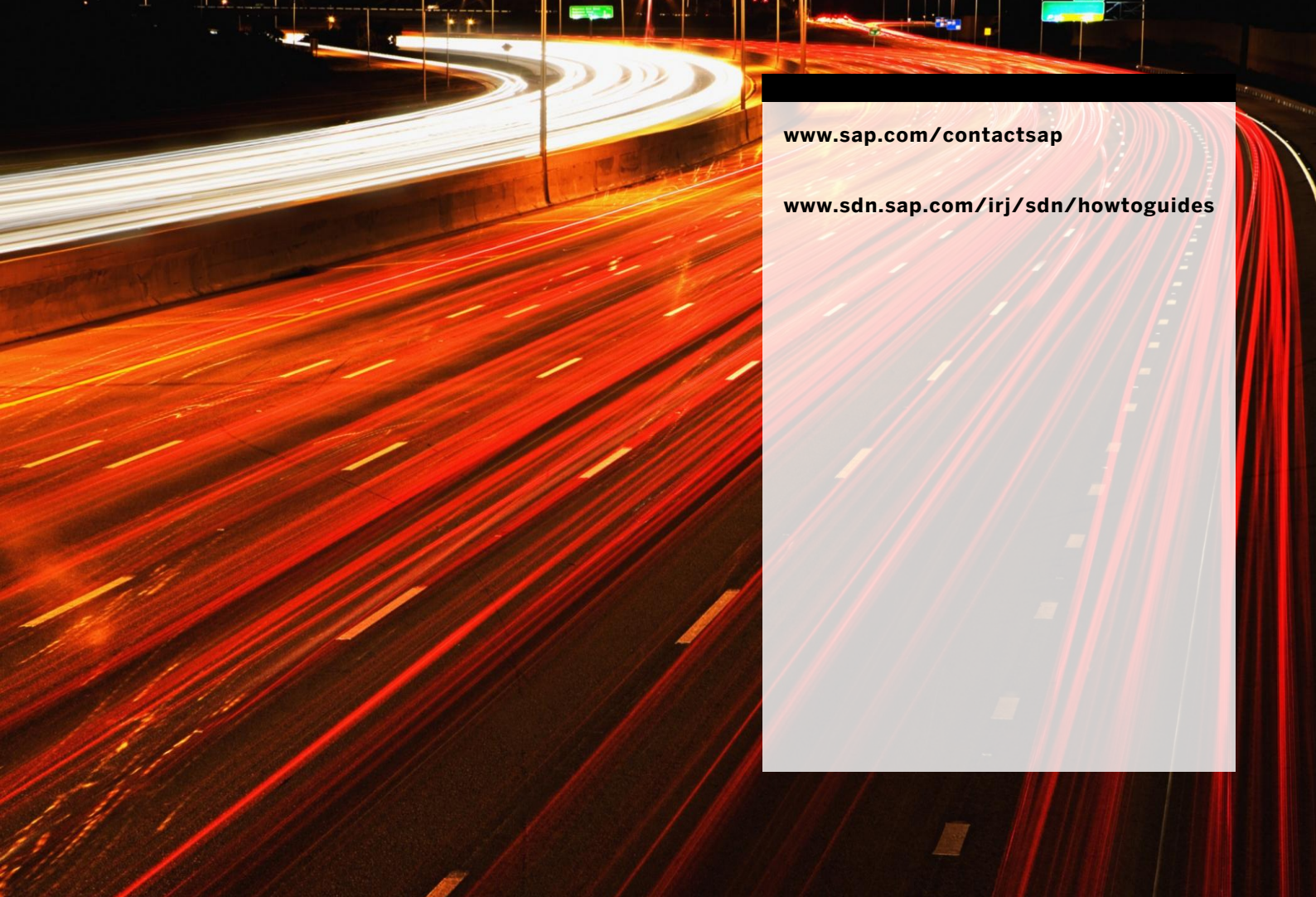
## The Application Library

SAP Fiori Apps that feature an approval process contain another component, called Application Library. While the basis for an approval-type SAP Fiori App is still the Application component, the views showing the details for a workflow item are contained in the Application Library.

The figure below illustrates the different components of an SAP Fiori Approval App further:



You may find the Application Library files in a subfolder structure of `/resources/sap/[component]/`. Application Libraries are also SAPUI5 libraries.



[www.sap.com/contactsap](http://www.sap.com/contactsap)

[www.sdn.sap.com/irj/sdn/howtoguides](http://www.sdn.sap.com/irj/sdn/howtoguides)



The Best-Run Businesses Run SAP™