How to Create a Very Simple FPM Application Based on BOL/GenIL

Applies to:

SAP Netweaver Release 7.31, SP 04

Summary

This tutorial describes how to create a very simple FPM application based on the BOL/GenIL Framework. In order to create this tutorial I used information specified in the BOL GenIL Programming Tutorial (https://wiki.wdf.sap.corp/wiki/display/WEBCUIF/GenIL+Programming+Tutorial+-+How+to+Guide)

Furthermore I used information specified in the ‚FPM@BOL HowTo' document (FPM@BOL_HowTo.doc).

Author: Matthias Hubert

Company: SAP

Created on: 11 September 2012

Author Bio

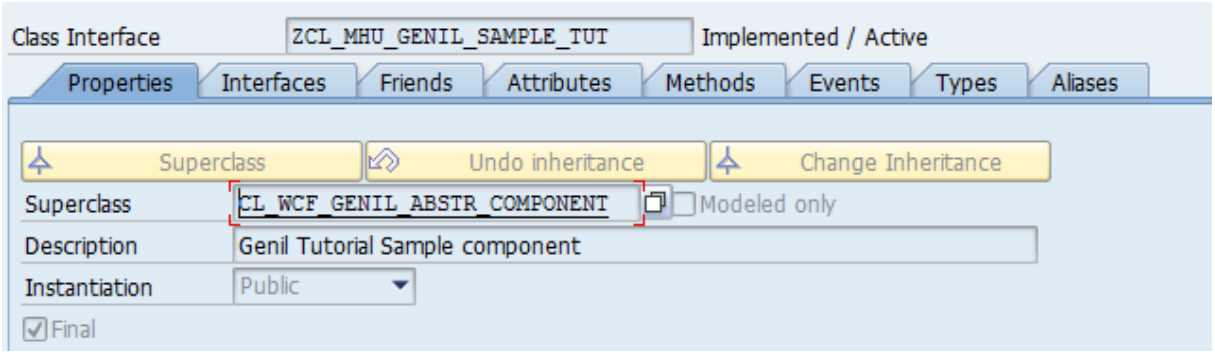
Name: Matthias Hubert
Company name: SAP
Field of work: ABAP application developer, Master Data Governance (MDG).

Table of Contents

1 Create GenIL Component

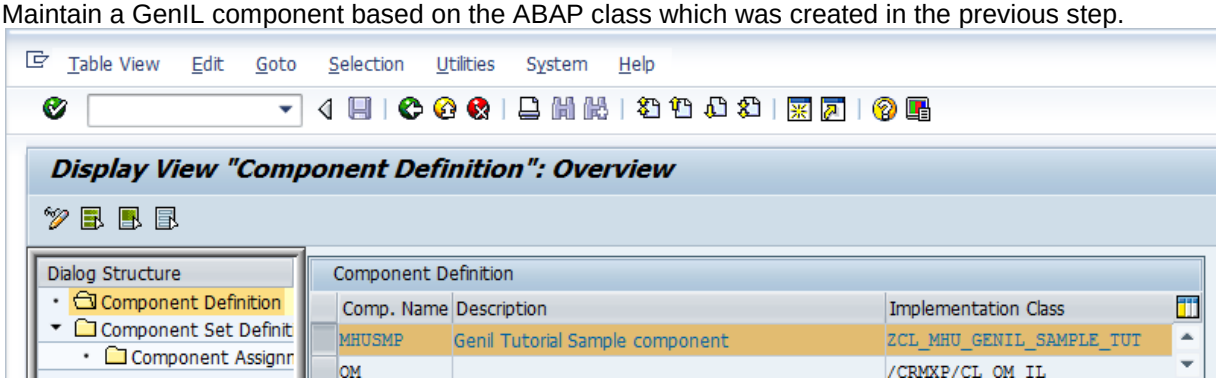
A GenIL Component corresponds to an ABAP class which specializes the class CL_WCF_GENIL_ABSTR_COMPONENT.

Example:

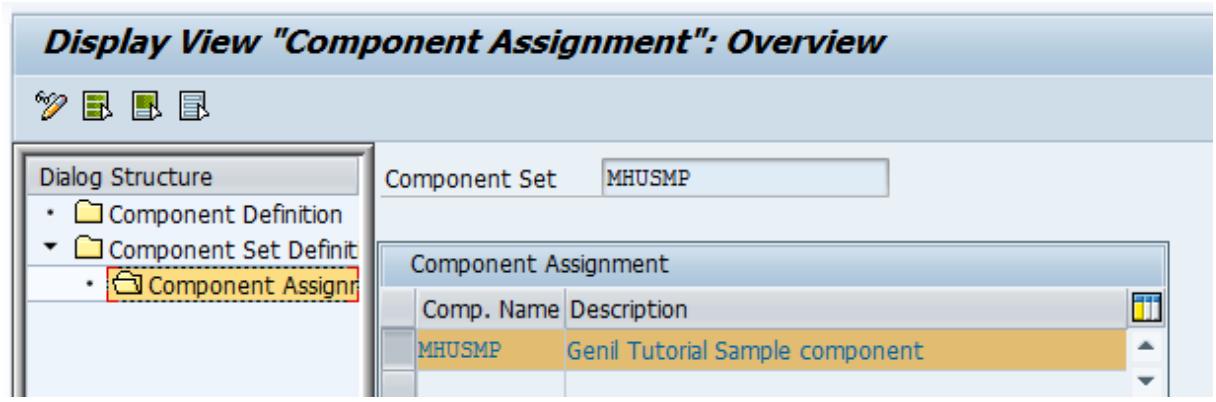


Go to transaction sm34 and open view cluster CRMVC_GIL_APPDEF.

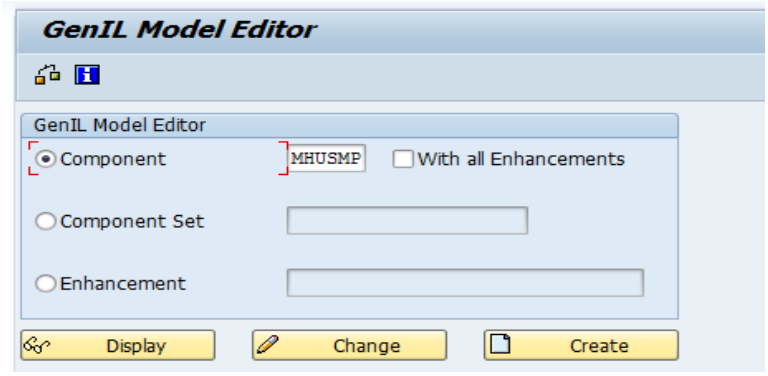
Maintain a GenIL component based on the ABAP class which was created in the previous step.



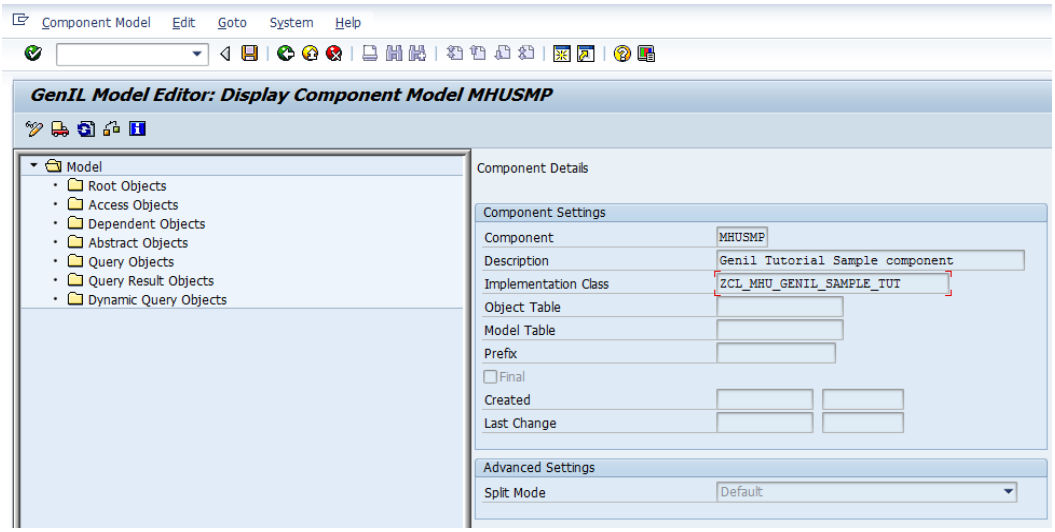
Assign the component to a component set.



Now you can test the GenIL component using transaction GENIL_MODEL_BROWSER.



Click on button ‚Display'.



The GenIL Model is displayed. As of now the model does not contain any objects. In the following steps objects will be added to model.

Summary:
A GenIL component implementation consists of two parts:

- An object model exposing the application entities and how they relate with each other
- The code which wraps the existing application API by implementing the GenIL component API. The GenIL implementation must comply to the model.

2 Create nodes in the object model

Create the following nodes in the object model of the GenIL component:

- MHUSMP_Account (Root Object)
- MHUSMP_AccountAddress (Dependent Object)
- MHUSMP_AccountQuery (Query Object)

You find a detailed description how to create the nodes in the BOL GenIL Programming Tutorial (Step 3).

After finishing the node creation the model should look like this:

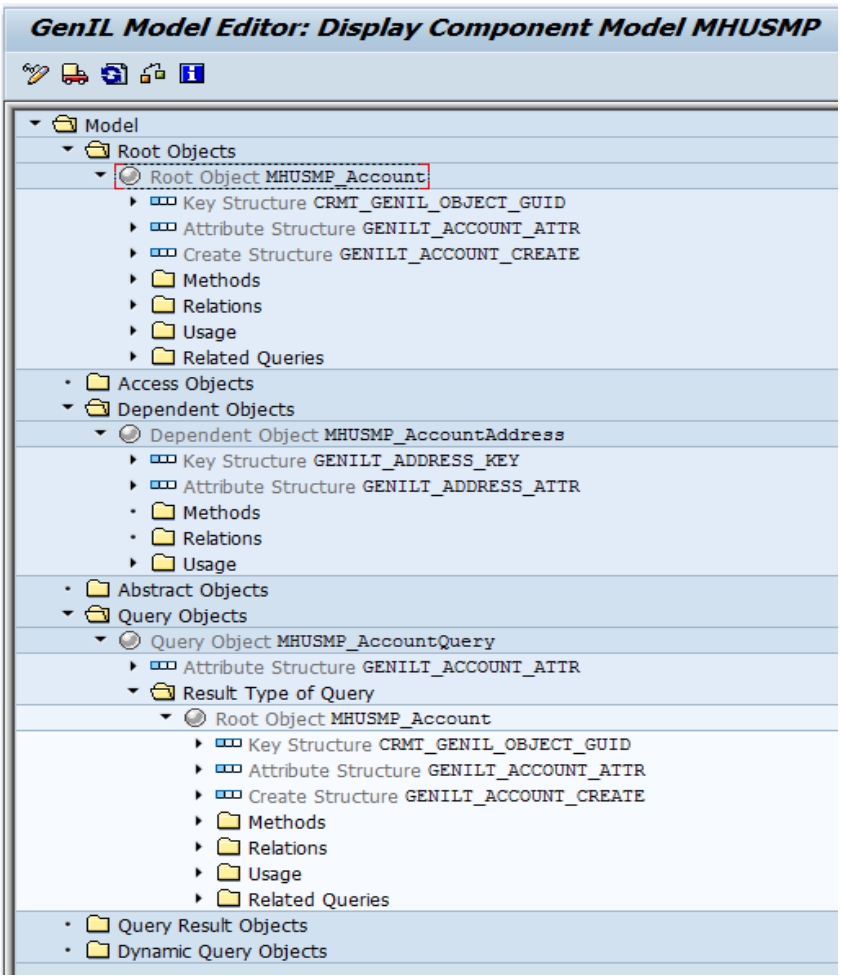


Abbildung 1: Object Model of GenIL component

3 Implement Query Object

Implement the Query Object (see Abbildung 1: Object Model of GenIL component) in order to implement a query for searching accounts. This means the method IF_GENIL_APPL_INTLAY~GET_QUERY_RESULT has to be implemented in the ABAP class of the GenIL component. You find a detailed description how to create the implementation in the BOL GenIL Programming Tutorial (Step 4).

4 Reading Root Object Data

Implement the method IF_GENIL_APPL_INTLAY~GET_OBJECTS in the ABAP class of the GenIL component. This method reads the Root Object data (see Abbildung 1: Object Model of GenIL component). You find a detailed description how to create the implementation in the BOL GenIL Programming Tutorial (Step 5).

5 Reading Related Object Data

Enhance the implementation of method IF_GENIL_APPL_INTLAY~GET_OBJECTS in the ABAP class of the GenIL component in order to read the related address data for an account. The address is a Dependent Object (see Abbildung 1: Object Model of GenIL component).

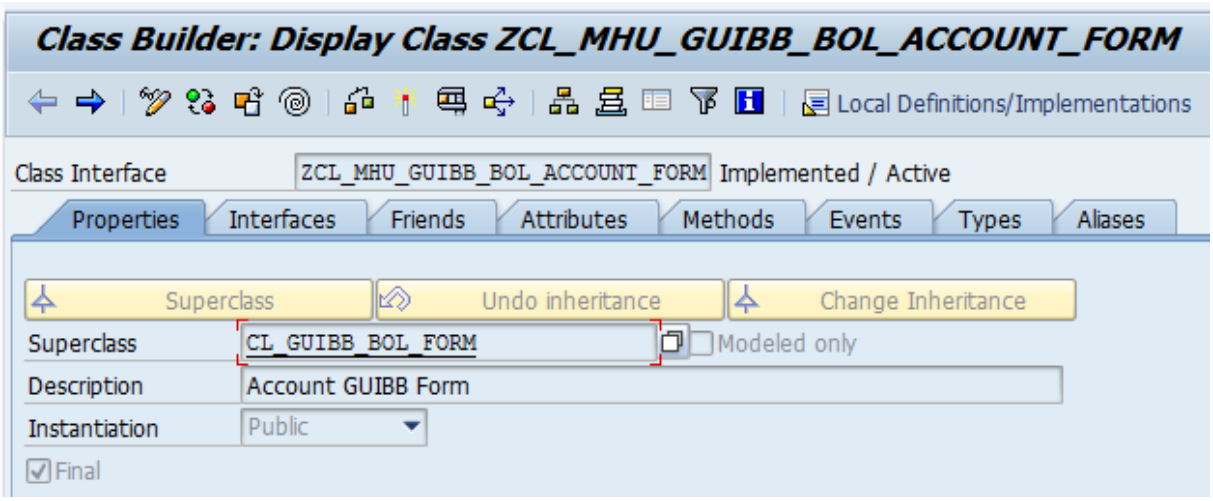
You find a detailed description how to enhance the implementation in the BOL GenIL Programming Tutorial (Step 6).

6 FPM application based on BOL/GenIL

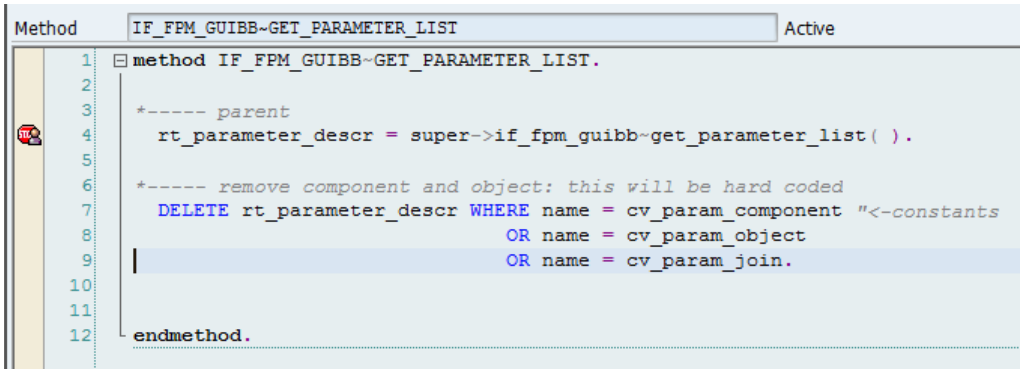
The application reads a single account using the GenIL component and displays the account data in a form-based layout using a Generic Form UIBB.

6.1 Derive feeder class from Generic BOL Form Feeder class

Create an ABAP class which inherits from the Generic BOL Form Feeder class (CL_GUIBB_BOL_FORM).



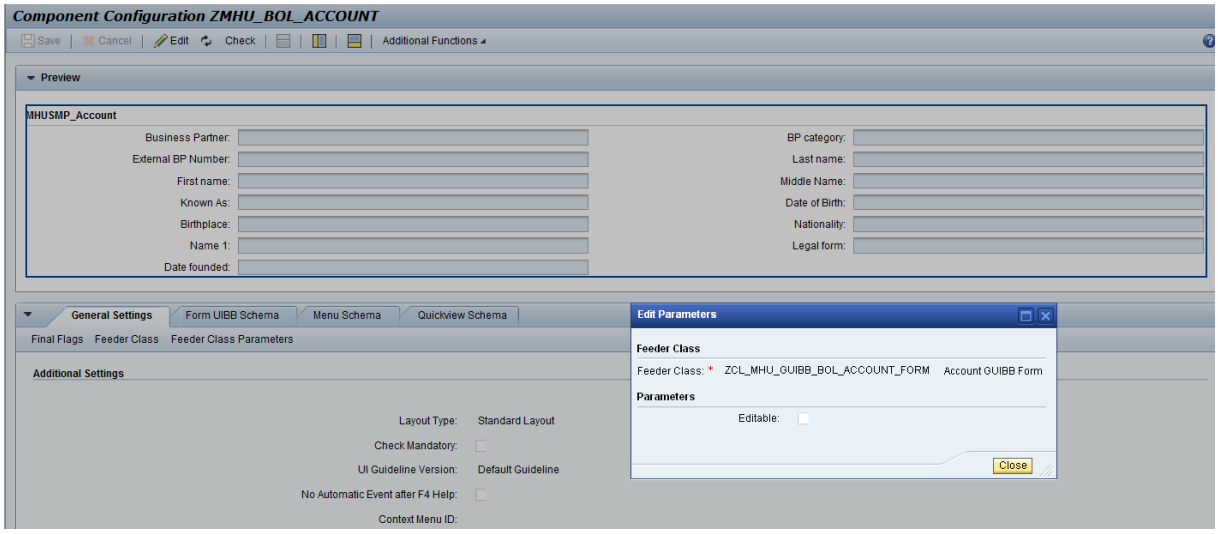
In this example the derived feeder class does not have any parameters. The GenIL component and the corresponding object of the object model (see Abbildung 1: Object Model of GenIL component) are specified in the coding. Therefore implement the following methods of the derived feeder class:



```
Method  EVALUATE_PARAMETERS                                   Active
 1  method EVALUATE_PARAMETERS.
 2  *CALL METHOD SUPER->EVALUATE_PARAMETERS
 3  *  EXPORTING
 4  *    IT_PARAMETER =
 5  *    .
 6
 7  *----- parent
 8    super->evaluate_parameters( it_parameter ).
 9
10  *---- hard coded component and object names
11    ms_object_key-component_name = 'MHUSMP'.   "Genil Component
12    ms_object_key-object_name = 'MHUSMP_Account'. "Object within Genil Component
13
14  endmethod.
```

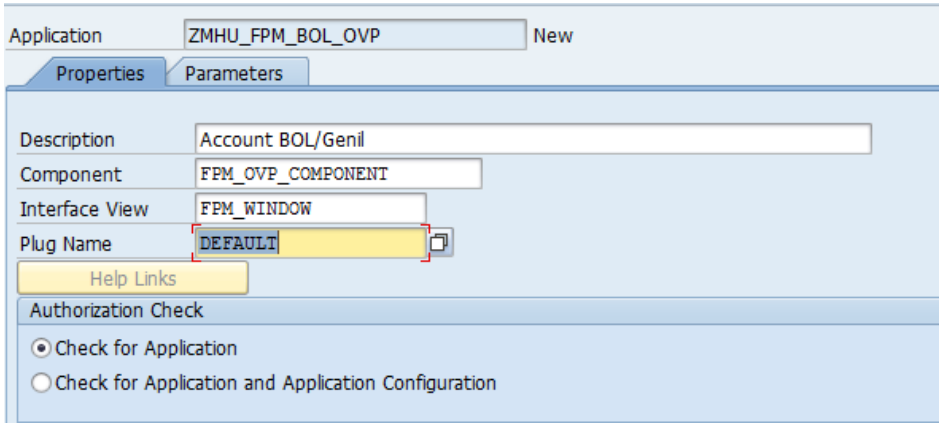
6.2 Create a component configuration for a FPM Form UIBB

Call the editor for a Web Dynpro ABAP Component Configuration and create a configuration for the Web Dynpro component FPM_FORM_UIBB.

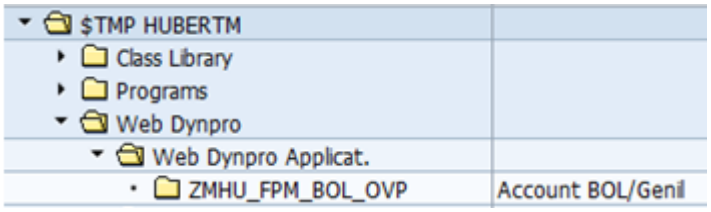


6.3 Create a Web Dynpro Application

In this example the Web Dynpro Application is based on an OVP Floorplan. Therefore create a Web Dynpro Application for Web Dynpro Component FPM_OVP_COMPONENT.

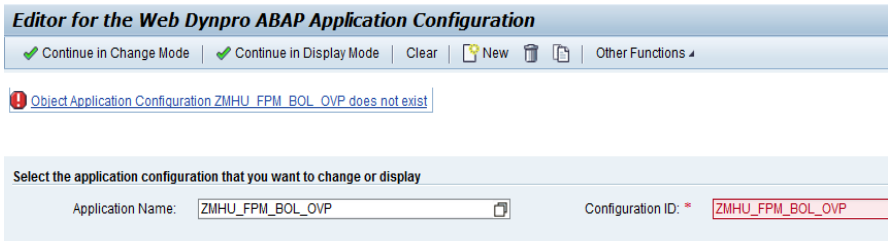


The WebDynpro Application is displayed in the Object Navigator:

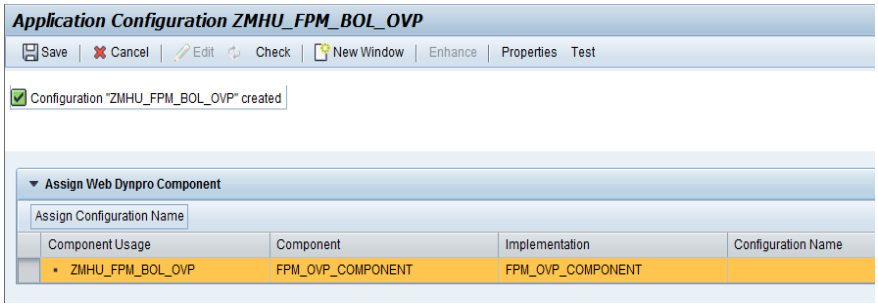


6.4 Create Application Configuration and assign Component Configuration

Now create an Application Configuration for Web Dynpro Application ZMHU_FPM_BOL_OVP.

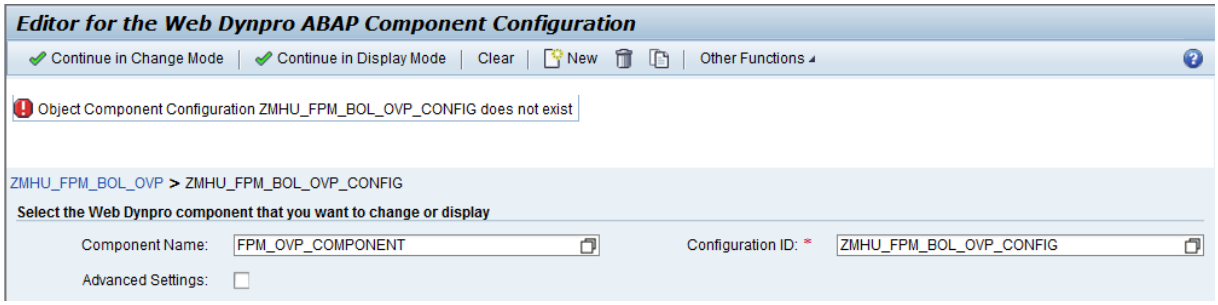


Then press ‚New' button.



Now click on button ‚Assign Configuration Name'.

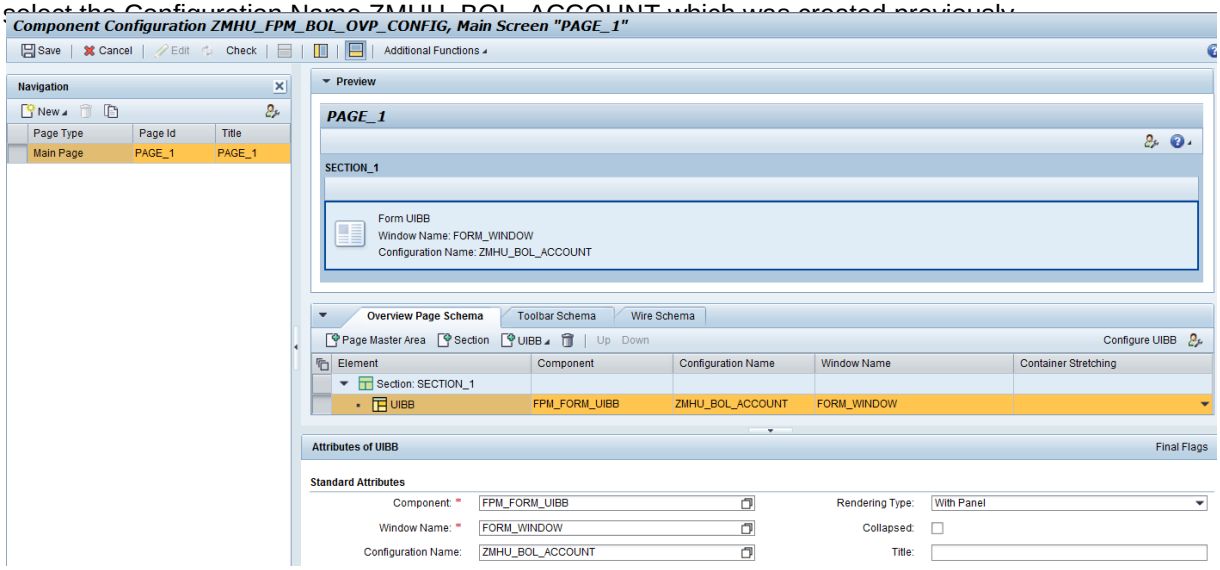
Enter: ZMHU_FPM_BOL_OVP_CONFIG (does not exist yet). Then double-click on the Configuration Name.



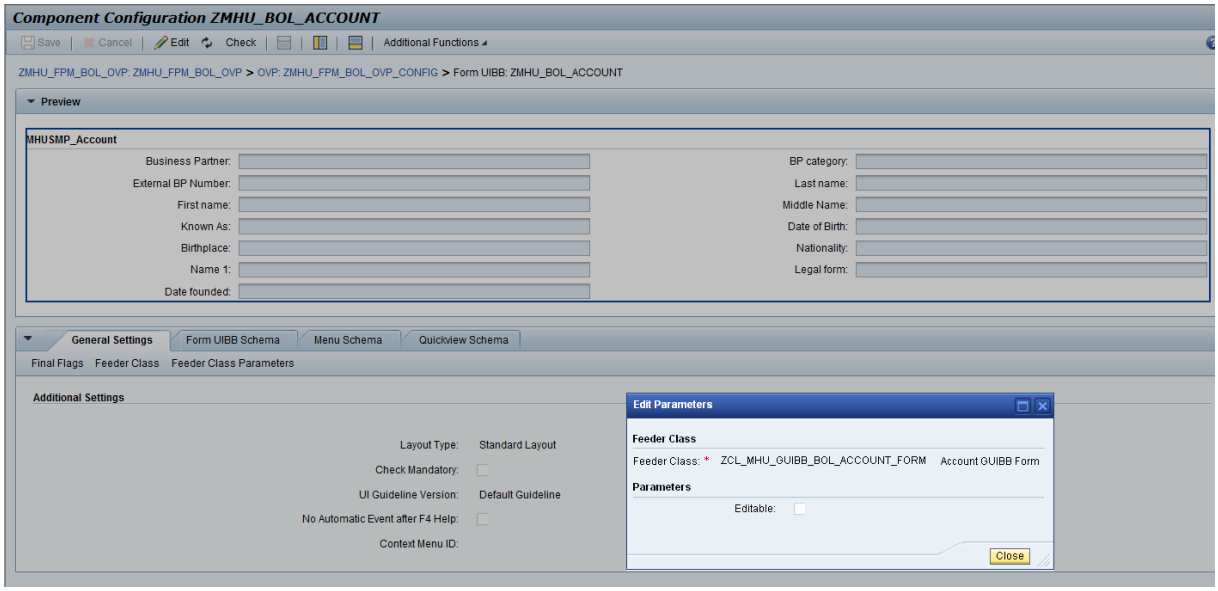
Then click on button ‚New' to create the configuration.

Then add a Form GUIBB.

Now select Component: FPM_FORM_UIBB, Window Name: FORM_WINDOW. Then use Search Help to select the Configuration Name ZMHU_BOL_ACCOUNT which was created previously.



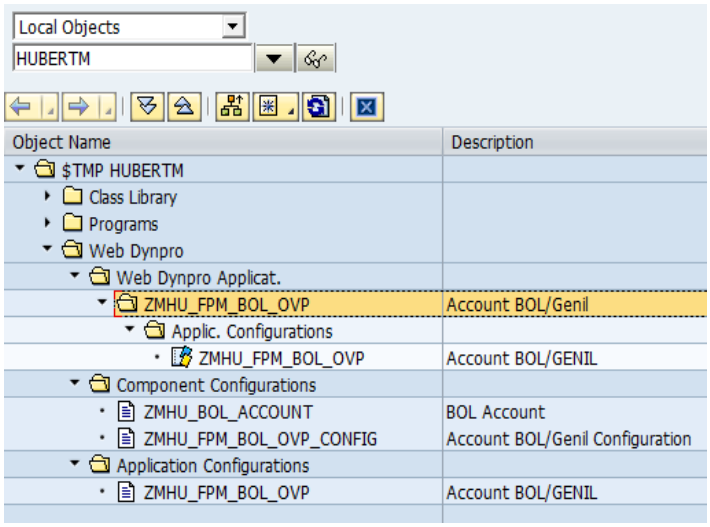
Then select UIBB and press button ‚Configure UIBB‘.



The fields are displayed according the attributes of the object ‚MHUSMP_Account‘ contained in Genil Component ‚MHUSMP_Account‘.

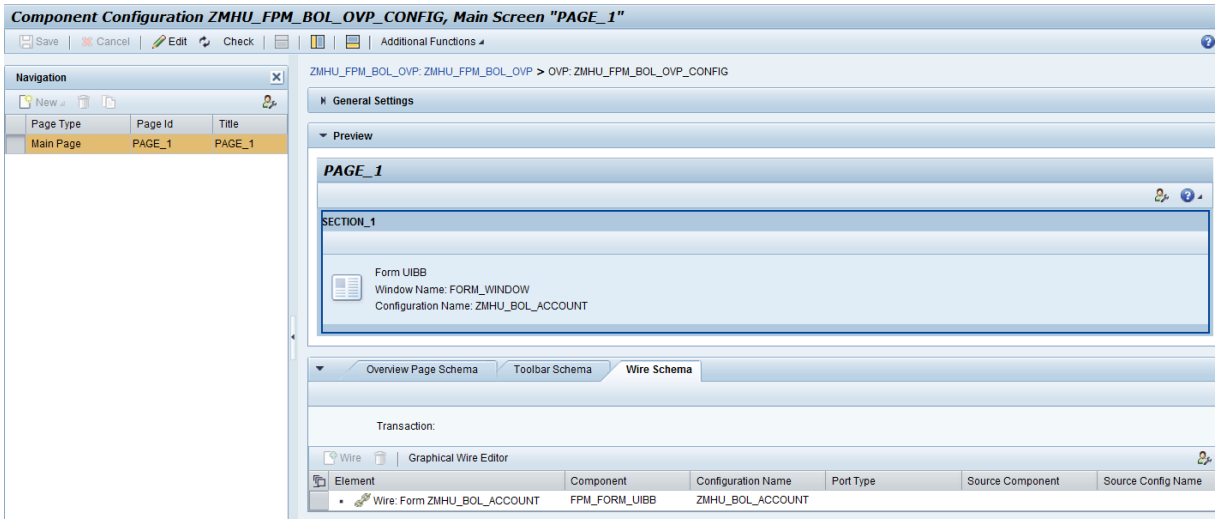
Then press ‚Save‘.

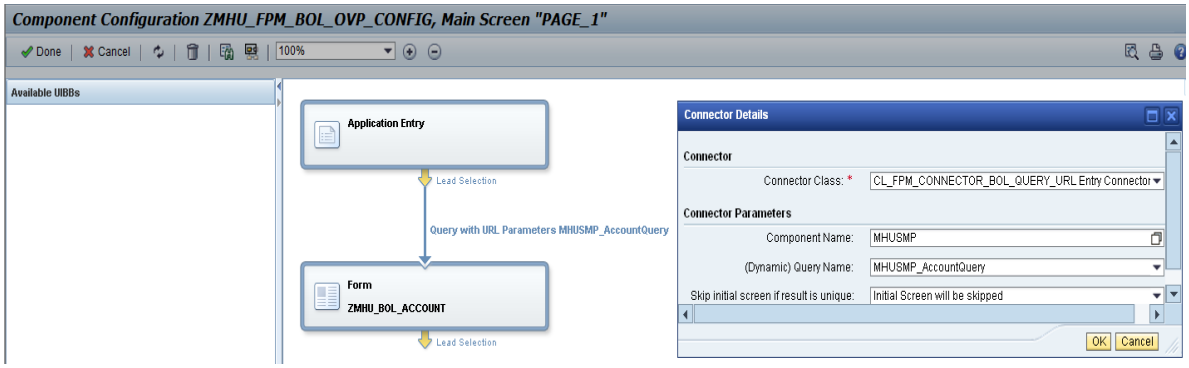
Verify in Object Navigator (se80) that the following objects are available:



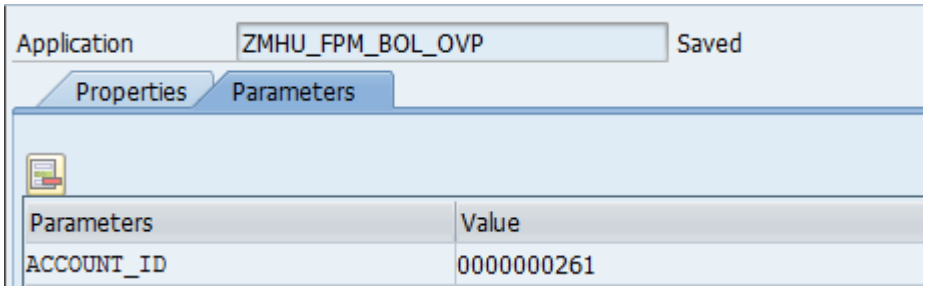
6.5 Define Wiring

In this example the application is started with a parameters (ACCOUNT_ID) determining the application data. This means the application data is fixed by URL parameters. A query is implicitly executed, i.e. invisible for the user, and the URL parameters supply the query parameters. This is done with the URL entry connector class CL_FPM_CONNECTOR_BOL_QUERY_URL. In the wire configuration of the floorplan component configuration, create a wire for the UIBB which shall we supplied with the result of the query. In this example the result of the query contains a single Account. You can use the Graphical Wire Editor maintain the required settings.



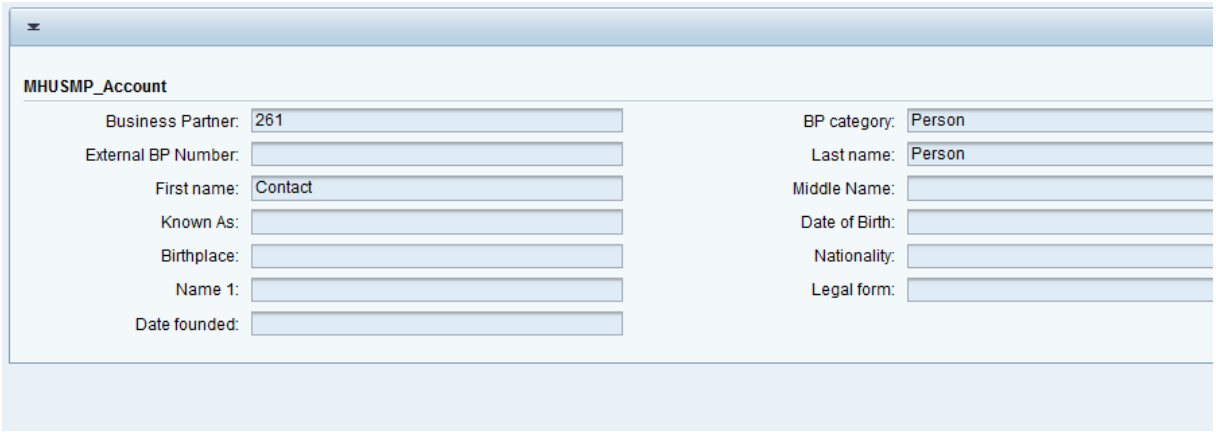


Now define an parameter ACCOUNT_ID for the Web Dynpro Application. The parameter contains the ID of the Account which shall be displayed by the application.



When the application ZMHU_FPM_BOL_OVP is started the account the parameter ACCOUNT_ID=0000000261 is passed in the URL to the application.

The Connector class CL_FPM_CONNECTOR_BOL_QUERY_URL reads the Account ID from the URL and executes the Query 'MHUSMP_AccountQuery' (see Abbildung 1: Object Model of GenIL component) using the Account ID as selection criteria. The query returns exactly one account which is then passed to the Form UIBB which displays the account data:



Related Content

FPM Developer's Handbook:
http://scn.sap.com/docs/DOC-30668

How to guide: BOL Programming:
http://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/e0ddd4e6-32f9-2c10-9d88-d993c4b0008d?QuickLink=index&overridelayout=true&46566035758209

FPM Overview:
http://scn.sap.com/docs/DOC-28804

Copyright