

Web Dynpro for ABAP: Tutorial 2 - BAPI Usage



SAP NetWeaver 04s



Copyright

© Copyright 2005 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, and Informix are trademarks or registered trademarks of IBM Corporation in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.






JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

Icons in Body Text

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help → General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

Typographic Conventions

Type Style	Description
<i>Example text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons 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	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
Example text	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
Example text	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	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.

Web Dynpro for ABAP: Tutorial 2 – BAPI Usage	5
Development Objectives	5
Procedure.....	6
Creating a Web Dynpro Component and a View	6
Creating a Service Call for BAPI <i>BAPI_FLIGHT_GETLIST</i>	6
Defining the Context Mapping	9
Defining Input Fields and a Button on the View	10
Defining an Action and Corresponding Action Handler.....	13
Defining the Table on the View	14
Embedding the View into the Window.....	16
Activation, Creation of a Web Dynpro Application and Execution	17
Result	17
SAP Online Help	18

Web Dynpro for ABAP: Tutorial 2 – BAPI Usage

Development Objectives

This exercise demonstrates the usage of BAPIs and how to implement a view with input fields and a result table.

You will create a Web Dynpro component `ZZ_00_BAPIFLIGHT` with one view `FLIGHTLISTVIEW`. The view contains several input fields, a button which triggers a search event and a table, in which the result list from the search is displayed. The component controller context refers to a BAPI structure and contains a service call for this BAPI (method `EXECUTE_BAPI_FLIGHT_GETLIST`). The component controller context structure is mapped to the view context of `FLIGHTLISTVIEW`, whereas the UI layout elements are bound to the context elements of the view.

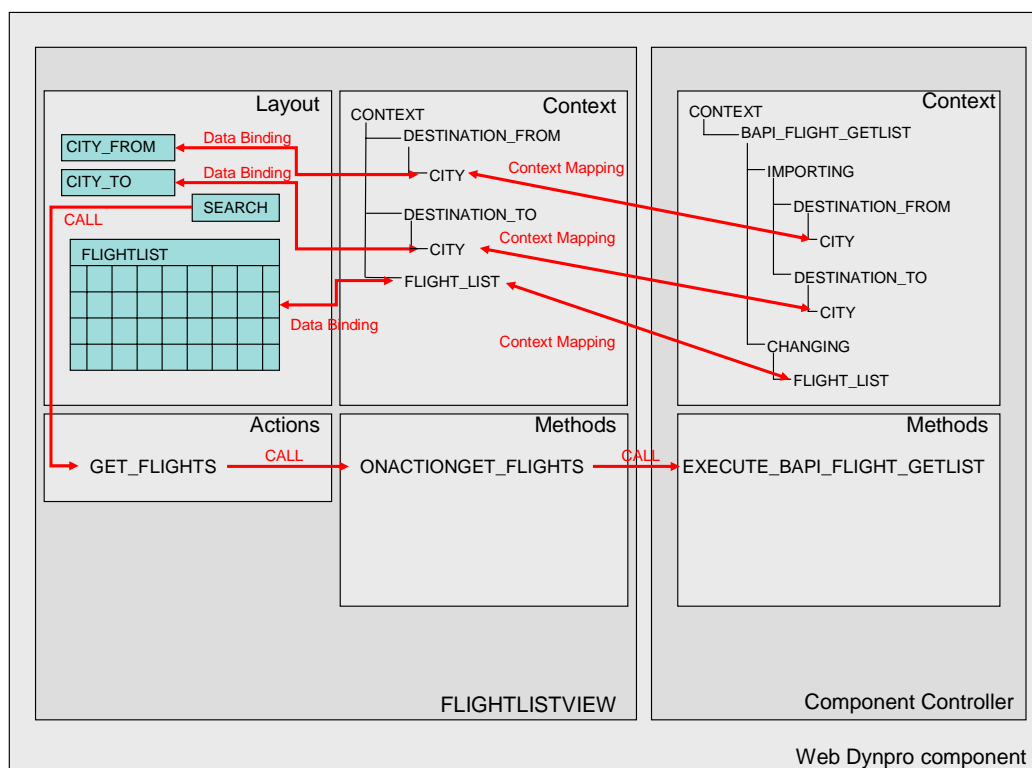


Figure 1: Schematic Representation of the Web Dynpro Component ZZ_00_BAPIFLIGHT



Procedures, which already were explained in detail in Tutorial 1, for example,

- how to create a Web Dynpro component, a view, or a Web Dynpro application,
- how to activate a Web Dynpro component, or
- how to execute a Web Dynpro application,

are not explained in detail in this tutorial. Therefore, we strongly recommend, that you first thoroughly work through Tutorial 1 before you start Tutorial 2.

Procedure

Creating a Web Dynpro Component and a View

1. Create a new Web Dynpro component with name ZZ_00_BAPIFLIGHT and assign it to package \$TMP (local object).
2. Create a view called *FLIGHTLISTVIEW*.
3. Save all the changes.

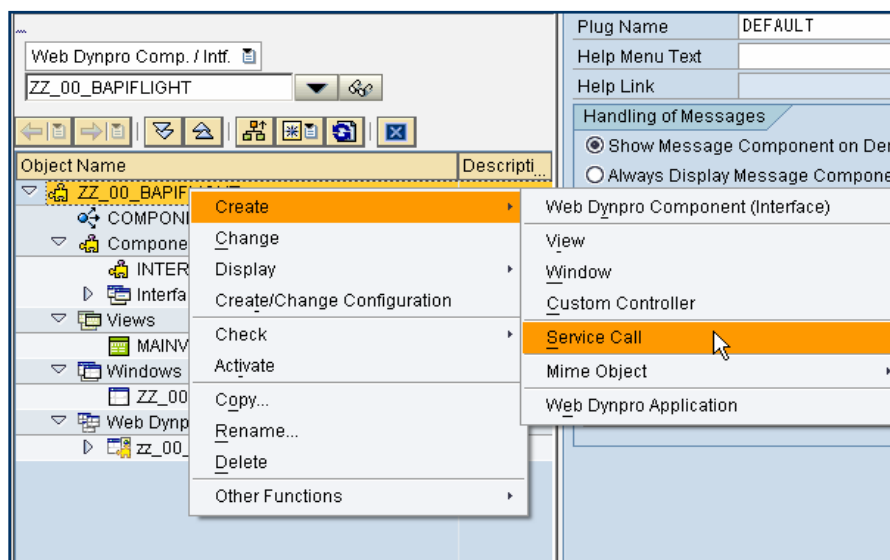
Creating a Service Call for BAPI *BAPI_FLIGHT_GETLIST*

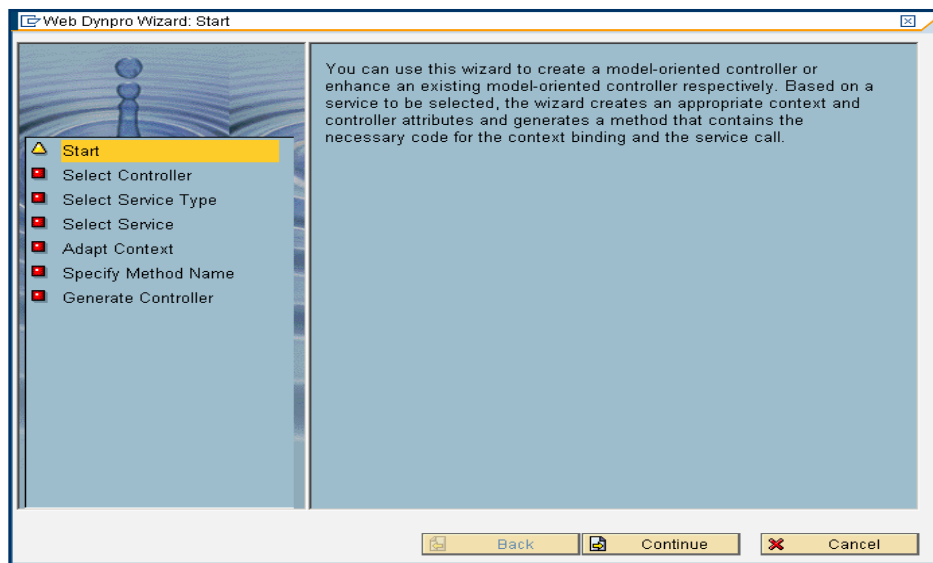
With the help of the service call function it is possible to call an existing function module from within a Web Dynpro component. To create a service call, you have an easy-to-use wizard at your disposal within the Web Dynpro tools in the ABAP Workbench.

Procedure

1. Starting the Wizard

To start the wizard, position the cursor on the Web Dynpro component to be edited in the object list at the left margin of the workbench window. Open its context menu and choose the entry *Create->Service Call*. The wizard is started and leads you through the creation process.





Press *Continue*.

2. Choice of Controller

On the second dialog window of the wizard, you can choose whether the service call is to be embedded in an existing controller or whether a new controller is to be created for this purpose.



Service calls can only always be embedded in global controllers – that is, in the component controller or in additionally created custom controllers. It is not possible, to embed service calls in view controllers.

- a. Select radio button *Use Existent Controller*
- b. Do not change the default entry for component: ZZ_00_BAPIFLIGHT
- c. Enter for controller COMPONENTCONTROLLER
- d. Press *Continue*.

3. Service Type and Service Selection

- a. You now select, which service type should be used for this service call. Select radio button *Function Module*. Leave Destination blank. Press *Continue*.
- b. Select the service: for Function Module enter *BAPI_FLIGHT_GETLIST*. Press *Continue*.

4. The Required Methods and Context Elements

On the two subsequent dialog windows, default values are listed for giving names to the context nodes and attributes required by the service call as well as to the required methods. The proposed names are based on the names of the embedded service, but you can change them as required. However, heed the respective notes in the corresponding dialog box.

- a. Adapt Context: Select from Nodes/Attributes Names *DESTINATION_FROM*, *DESTINATION_TO* and *FLIGHT_LIST*. Press *Continue*.

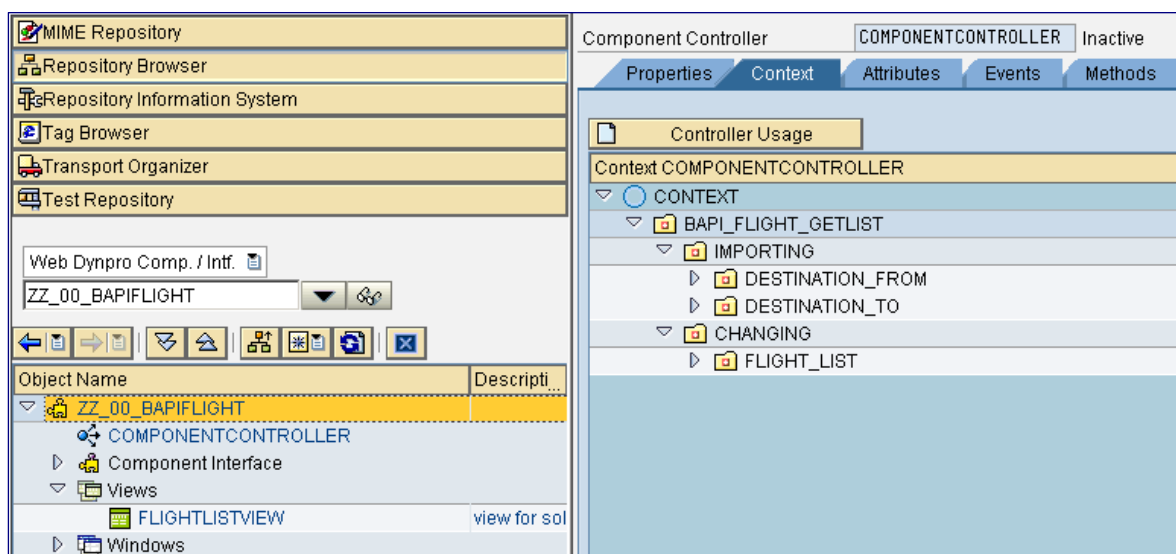
- b. Specify Method Name: leave all entries as provided:
 Component: `ZZ_00_BAPIFLIGHT`
 Controller: `COMPONENTCONTROLLER`
 Method: `EXECUTE_BAPI_FLIGHT_GETLIST`
 Press *Continue*.

5. Completing the Choice

When you have confirmed the last dialog box, the generation is triggered. Afterwards you now have the required methods and contexts at your disposal for using them within your Web Dynpro component.

Result

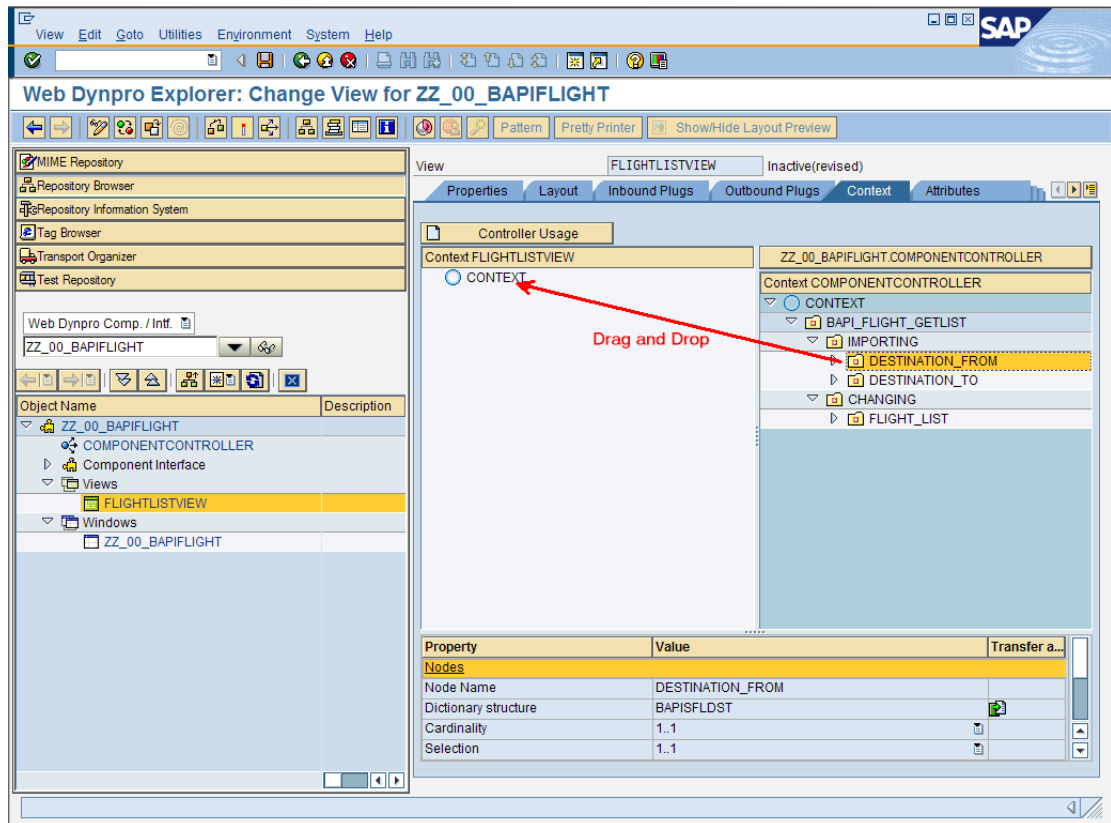
The component controller context now contains the corresponding context nodes for the BAPI call.



Furthermore, in the method list there is a new method `EXECUTE_BAPI_FLIGHT_GETLIST`, which contains the coding to read the context nodes `DESTINATION_FROM` and `DESTINATION_TO` and their underlying context attributes, calls the BAPI and assigns the resulting table to context node `FLIGHT_LIST`.

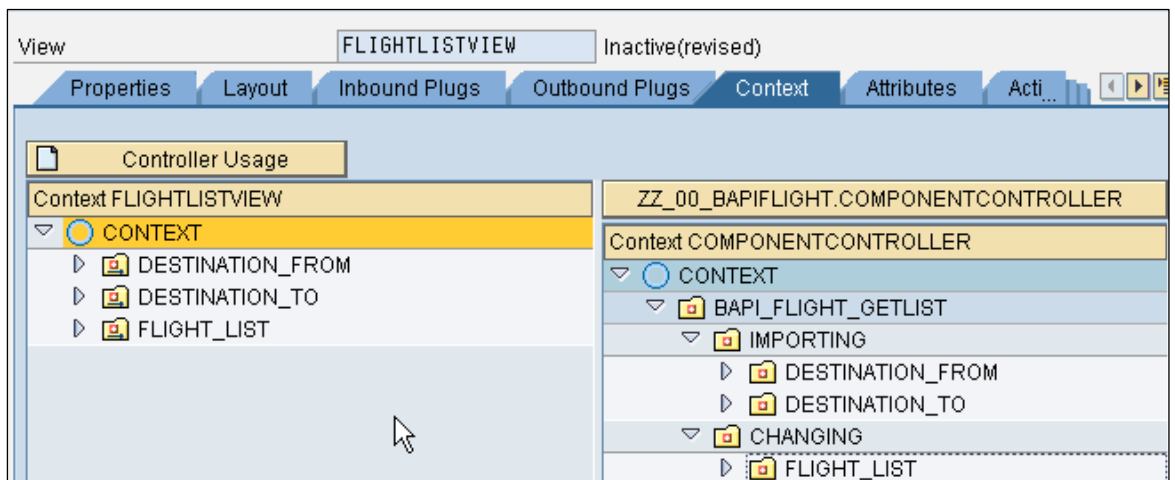
Defining the Context Mapping

1. Open view FLIGHTLISTVIEW and switch to tab *Context*. Map context nodes DESTINATION_FROM, DESTINATION_TO and FLIGHT_LIST to the view context of FLIGHTLISTVIEW using Drag and Drop.



Confirm the dialogues whether you want to copy or map the different context nodes.

The result should look like this:



2. Save your changes.

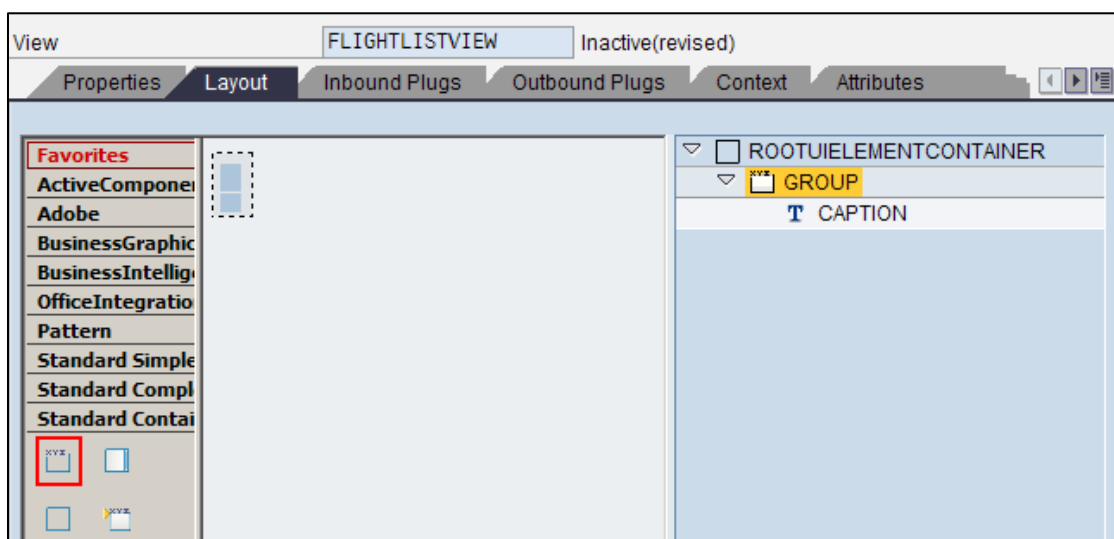
You have now mapped the component controller context elements to the view context of FLIGHTLISTVIEW.

Defining Input Fields and a Button on the View

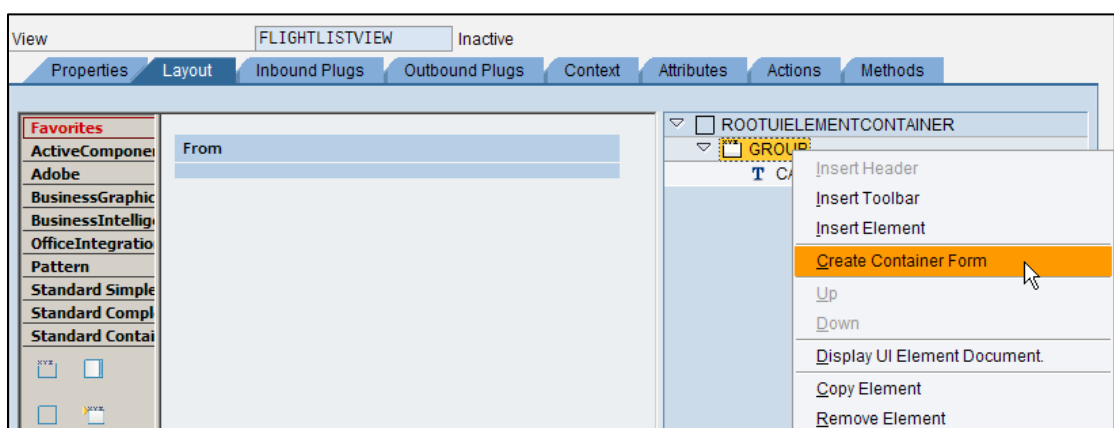
1. Switch to tab *Layout* of view FLIGHTLISTVIEW.
2. Select *Standard Container* from UI Elements Library and drag and drop the Group icon to the View Designer. The new UI element will be named *GROUP*.

Set the following properties for *GROUP*:

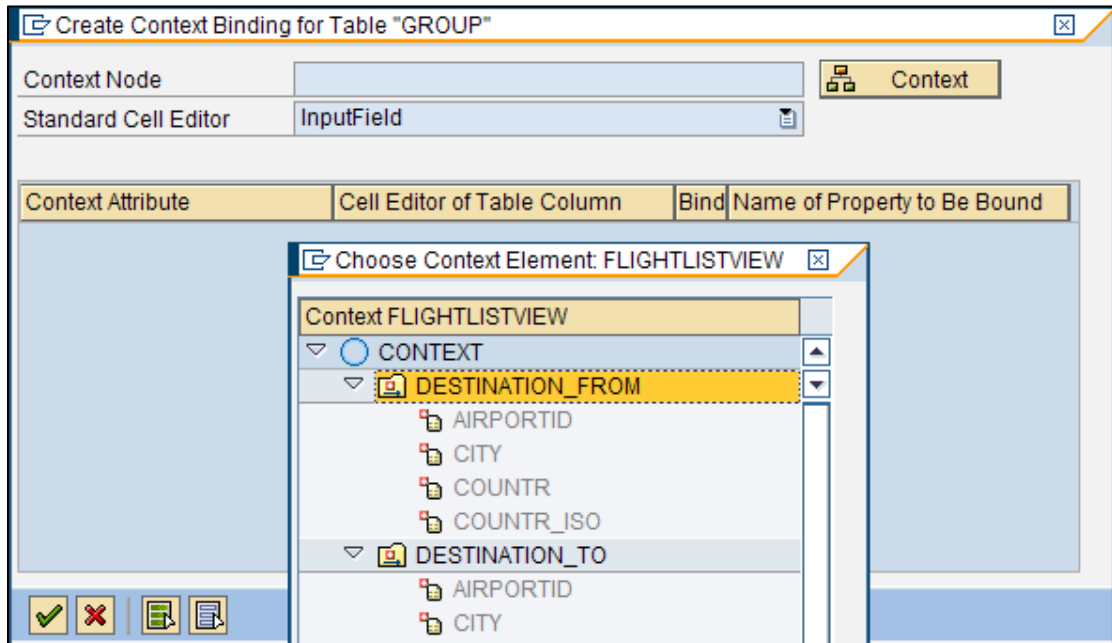
GROUP properties	
Caption → Text	From
Width	100%



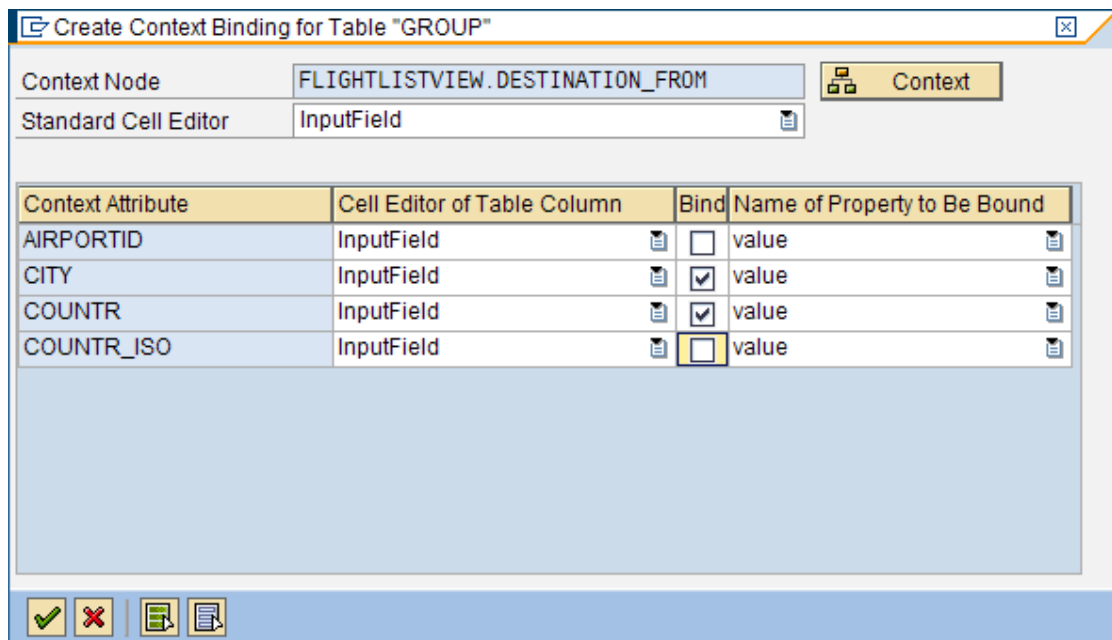
3. Right-click on *GROUP* and choose *Create Container Form*.



- Click on the *Context* button and create a binding with context node *DESTINATION_FROM* by double-clicking it.



- Select the attributes *CITY* and *COUNTR* and leave the Standard Cell Editor as *INPUTFIELD*.

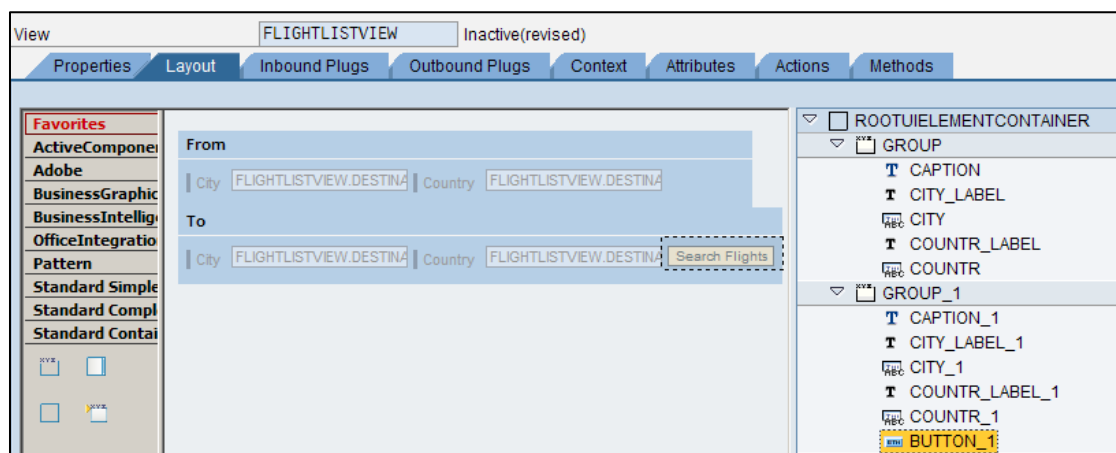


Repeat the same procedure (2-5) to create an UI element *GROUP_1*, bind it with context node *DESTINATION_TO* and select again the attributes *CITY* and *COUNTR*.

Set the following properties for *GROUP_1*:

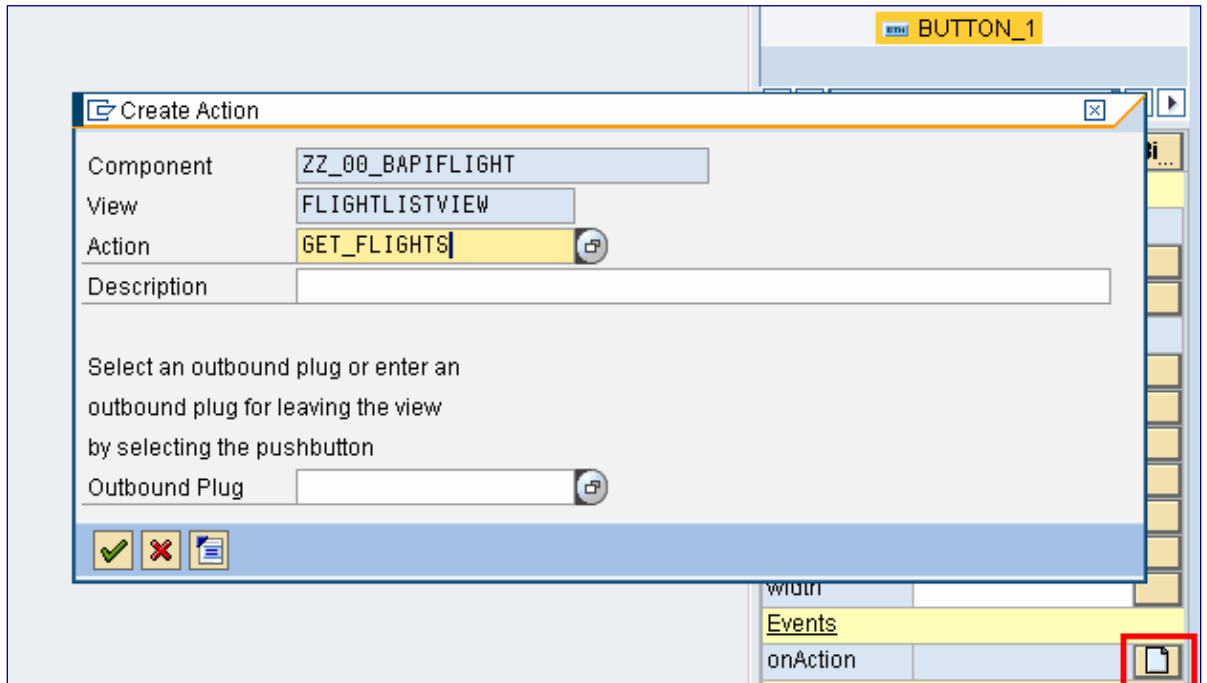
GROUP_1 properties	
Caption1 → Text	To
Width	100%

6. Additionally, create a button within *GROUP_1*, by right clicking the *GROUP_1* element below the *ROOTUIELEMENTCONTAINER* and select *Insert Element* from context menu. Choose type Button as UI element and name it *BUTTON_1*. The button will be placed in the group container.
7. Set the property *Text* of *BUTTON_1* to value *Search Flights*.



Defining an Action and Corresponding Action Handler

1. Create an action `GET_FLIGHTS` which is triggered by the Web Dynpro event `onAction`. In properties table of `BUTTON_1`, select the event property `onAction`. Click the icon on the right corner (empty page) to define a new action.



A double-click on event `GET_FLIGHTS` will generate a skeleton for method `ONACTIONGET_FLIGHTS`.

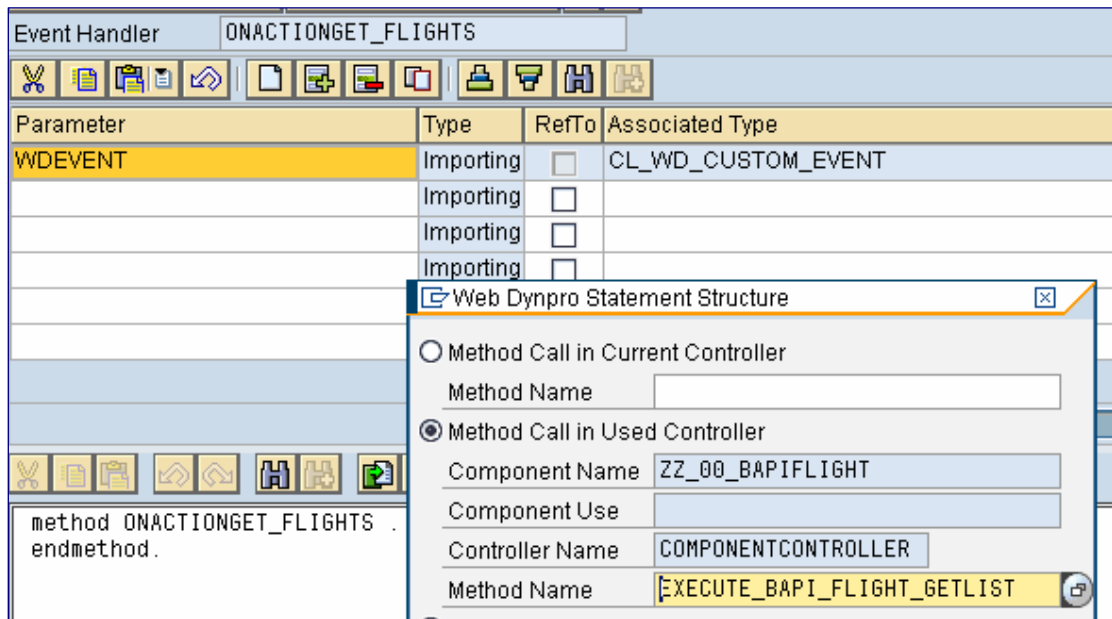
width		
Events		
onAction	GET_FLIGHTS	
Layout Data (FlowData)		
cellDesign	padless	
vGutter	none	

The code editor will open automatically.

2. Now the event handler for action `GET_FLIGHTS`, which is method `ONACTIONGET_FLIGHTS`, needs to be modified to make a call to the controller method `EXECUTE_BAPI_FLIGHT_GETLIST`. In order to do this, invoke the Web Dynpro Code Wizard and choose option *Method Call in Used Controller*.

Enter the following data (you may use F4 Help to get the possible entries).

Component Name: ZZ_00_BAPIFLIGHT
 Controller Name: COMPONENTCONTROLLER
 Method Name: EXECUTE_BAPI_FLIGHT_GETLIST



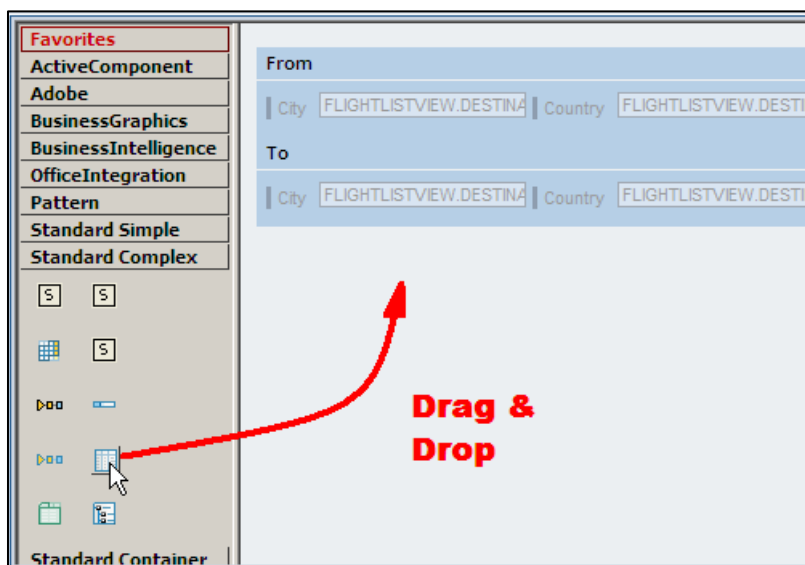
3. Select *Continue* (Enter).

The generated coding will look like this.

```
method ONACTIONGET_FLIGHTS .
    wd_Comp_Controller->Execute_Bapi_Flight_Getlist( ).
endmethod.
```

Defining the Table on the View

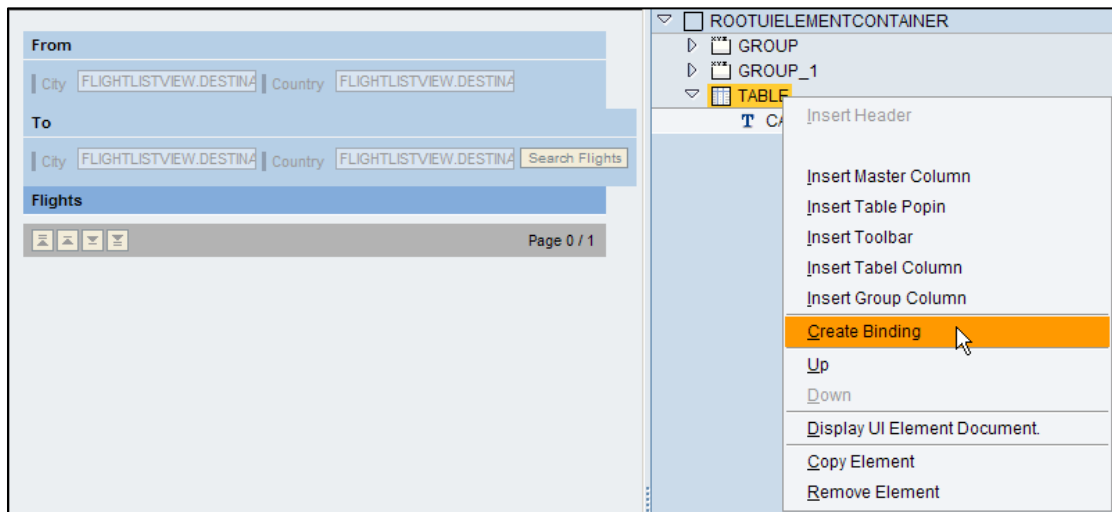
1. Switch to the *Layout* tab of the view and create a table *TABLE* using **Standard Complex** from UI Elements Library in View Designer (Drag and Drop).



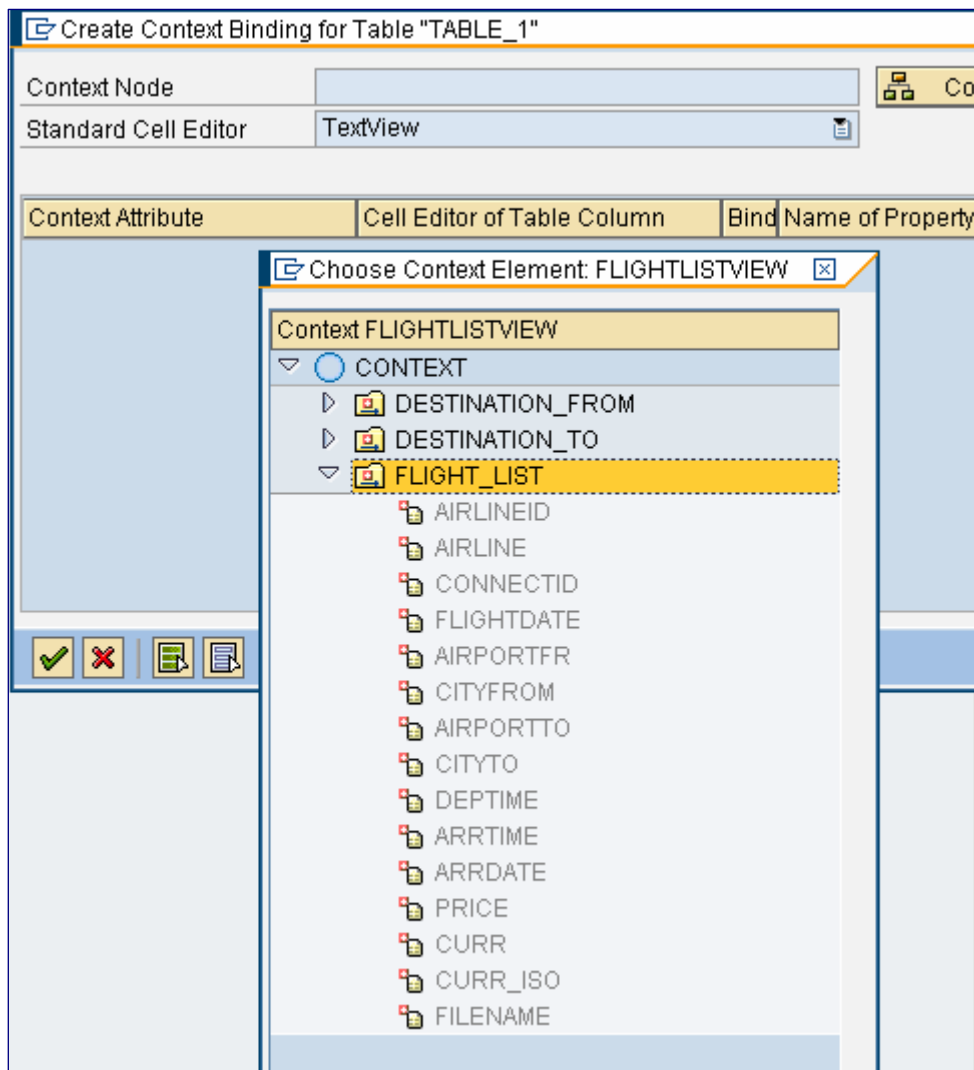
2. Set the following properties for the new element TABLE:

Table properties	
Caption_2 → Text	Flights
Width	100%

3. Create the data binding of UI element *TABLE* with context node *FLIGHT_LIST*.



Standard Cell Editor should be of type *TEXTVIEW*. Activate binding for all context attributes.

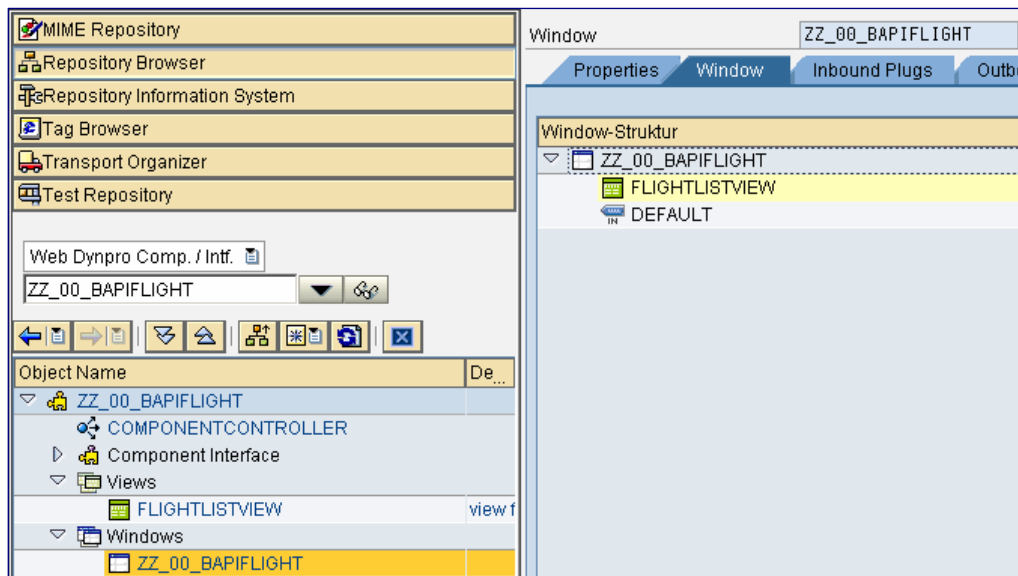


4. Press button *Continue (Enter)* and save all the changes.

Embedding the View into the Window

Embed the view FLIGHTLISTVIEW into Window ZZ_00_BAPIFLIGHT.

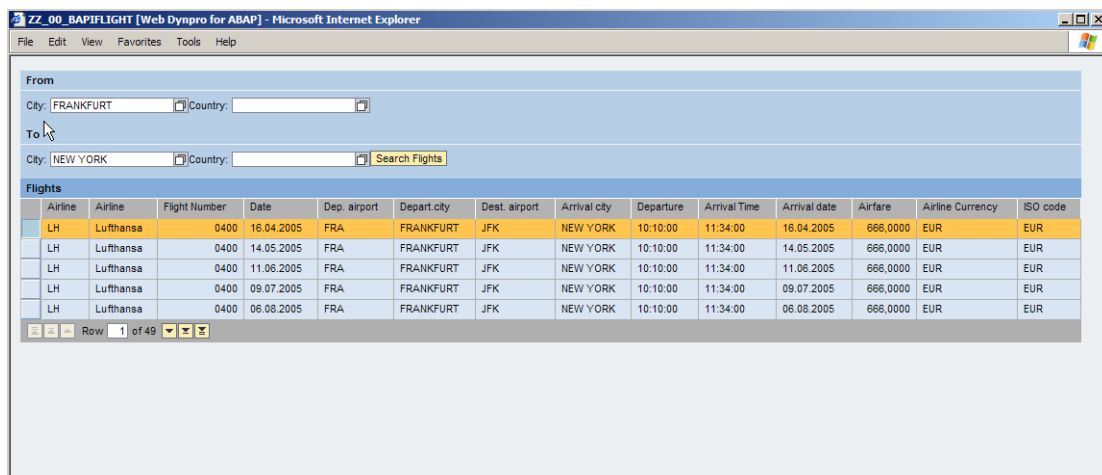
1. Open in the Object Navigator tree the Window structure and select the window ZZ_00_BAPIFLIGHT.
2. Open the view structure and drag and drop the view FLIGHTLISTVIEW inside the window structure on the right hand side.
3. Open the window structure on the right hand side and you will see the embedded view FLIGHTLISTVIEW.
4. Save your changes.



Activation, Creation of a Web Dynpro Application and Execution

1. Activate **all objects** of Web Dynpro component *ZZ_00_BAPIFLIGHT*.
2. Create the Web Dynpro application *ZZ_00_BAPIFLIGHT* and assign it to package *\$TMP* (local object).
3. Run your application.

The result should look like this.



Result

You have now created a simple Web Dynpro component which uses a BAPI call for data collection. You have seen how to use the service call wizard to easily create a service call from a function module.

SAP Online Help

More information on Web Dynpro for ABAP can be found at the SAP Help Portal under the short link

http://help.sap.com/saphelp_nw04s/helpdata/en/77/3545415ea6f523e10000000a155106/frameaset.htm or via path help.sap.com → Documentation → SAP NetWeaver → SAP NetWeaver

2004s → English → SAP NetWeaver Library → SAP NetWeaver by Key Capability →

Application Platform by Key Capability → ABAP Technology → UI Technology → Web UI

Technology → Web Dynpro for ABAP.